EP 30 8 60

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 715 247 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
05.06.1996 Bulletin 1996/23

(51) Int Cl.[6]: **G06F 1/00**

(21) Application number: **95308422.5**

(22) Date of filing: **23.11.1995**

(84) Designated Contracting States:
DE FR GB

(30) Priority: **23.11.1994 US 344760**

(71) Applicant: **XEROX CORPORATION**
Rochester New York 14644 (US)

(72) Inventors:
- **Stefik, Mark J.**
  Woodside, California 94062 (US)
- **Pirolli, Peter L. T.**
  El Cerrito, California 94530 (US)

(74) Representative: **Goode, Ian Roy**
Rank Xerox Ltd
Patent Department
Parkway
Marlow Buckinghamshire SL7 1YL (GB)

(54) System for controlling the distribution and use of digital works using digital tickets

(57) A system for controlling the distribution and use of digital works (101) using digital tickets. In the present invention, a "digital ticket" is used to entitle the ticket holder to exercise some usage right (102) with respect to a digital work. Usage rights are used to define how a digital work may be used or distributed. Each usage right may specify a digital ticket which must be present before the right may be exercised. Digital works are stored in repositories which enforce digital works usage rights (105) when usage of a digital work is requested by a requesting repository (103,104). Each repository has a "generic ticket agent" which punches tickets. In some instances only the generic ticket agent is necessary. In other instances, punching by a "special ticket agent" residing on another repository may be needed.

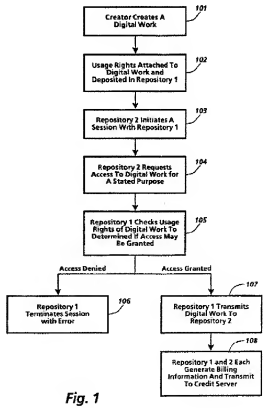

Fig. 1

EP 0 715 247 A1

<DOCID: <EP 0715247A1_I_>

Description

The present invention relates to the field of distribution and usage rights enforcement for digitally encoded works.

A fundamental issue facing the publishing and information industries as they consider electronic publishing is how to prevent the unauthorized and unaccounted distribution or usage of electronically published materials. Electronically published materials are typically distributed in a digital form and recreated on a computer based system having the capability to recreate the materials. Audio and video recordings, software, books and multimedia works are all being electronically published. Companies in these industries receive royalties for each accounted for delivery of the materials, e.g. the sale of an audio CD at a retail outlet. Any unaccounted distribution of a work results in an unpaid royalty (e.g. copying the audio recording CD to another digital medium.)

The ease in which electronically published works can be "perfectly" reproduced and distributed is a major concern. The transmission of digital works over networks is commonplace. One such widely used network is the Internet. The Internet is a widespread network facility by which computer users in many universities, corporations and government entities communicate and trade ideas and information. Computer bulletin boards found on the Internet and commercial networks such as CompuServ and Prodigy allow for the posting and retrieving of digital information. Information services such as Dialog and LEXIS/NEXIS provide databases of current information on a wide variety of topics. Another factor which will exacerbate the situation is the development and expansion of the National Information Infrastructure (the NII). It is anticipated that, as the NII grows, the transmission of digital works over networks will increase many times over. It would be desirable to utilize the NII for distribution of digital works without the fear of widespread unauthorized copying.

The most straightforward way to curb unaccounted distribution is to prevent unauthorized copying and transmission. For existing materials that are distributed in digital form, various safeguards are used. In the case of software, copy protection schemes which limit the number of copies that can be made or which corrupt the output when copying is detected have been employed. Another scheme causes software to become disabled after a predetermined period of time has lapsed. A technique used for workstation based software is to require that a special hardware device must be present on the workstation in order for the software to run, e.g., see US-A-4,932,054 entitled "Method and Apparatus for Protecting Computer Software Utilizing Coded Filter Network in Conjunction with an Active Coded Hardware Device. " Such devices are provided with the software and are commonly referred to as dongles.

Yet another scheme is to distribute software, but which requires a "key" to enable its use. This is employed in distribution schemes where "demos" of the software are provided on a medium along with the entire product. The demos can be freely used, but in order to use the actual product, the key must be purchased. These schemes do not hinder copying of the software once the key is initially purchased.

It is an object of the present invention to provide an improved system and method for controlling the use and distribution of digital works.

The invention accordingly provides a system and method as claimed in the accompanying claims.

A system for controlling the distribution and use of digital works using digital tickets is disclosed. A ticket is an indicator that the ticket holder has already paid for or is otherwise entitled to some specified right, product or service. In the present invention, a "digital ticket" is used to enable the ticket holder to exercise usage rights specifying the requirement of the digital ticket. Usage rights are used to define how a digital work may be used or distributed. Specific instances of usage rights are used to indicate a particular manner of use or distribution. A usage right may specify a digital ticket which must be present before the right may be exercised. For example, a digital ticket may be specified in a Copy right of a digital work, so that exercise of the Copy right requires the party that desires a copy of the digital work be in possession of the necessary digital ticket. After a copy of the digital work is successfully sent to the requesting party, the digital ticket is "punched" to indicate that a copy of the digital work has been made. When the ticket is "punched" a predetermined number of times, it may no longer be used.

Digital works are stored in repositories. Repositories enforce the usage rights for digital works. Each repository has a "generic ticket agent" which punches tickets. In some instances only the generic ticket agent is necessary. In other instances, punching by a "special ticket agent" residing on another repository may be desired. Punching by a "special ticket agent" enables greater security and control of the digital work. For example, it can help prevent digital ticket forgery. Special ticket agents are also useful in situations where an external database needs to be updated or checked.

A digital ticket is merely an instance of a digital work. Thus, a digital ticket may be distributed among repositories in the same fashion as other digital works.

A digital ticket may be used in many commercial scenarios such as in the purchase of software and prepaid upgrades. A digital ticket may also be used to limit the number of times that a right may be exercised. For example, a user may purchase a copy of a digital work, along with the right to make up to 5 Copies. In this case, the Copy right would have associated therewith a digital ticket that can be punched up to 5 times. Other such commercial scenarios will become apparent from the detailed description.

EDOCID: <EP 0715247A1 I >

A system and method in accordance with the invention will now be described, by way of example, with reference to the accompanying drawings, in which:-

Figure 1 is a flowchart illustrating a simple instantiation of the operation of the currently preferred embodiment of the present invention.

Figure 2 is a block diagram illustrating the various repository types and the repository transaction flow between them in the currently preferred embodiment of the present invention.

Figure 3 is a block diagram of a repository coupled with a credit server in the currently preferred embodiment of the present invention.

Figures 4a and 4b are examples of rendering systems as may be utilized in the currently preferred embodiment of the present invention.

Figure 5 illustrates a contents file layout for a digital work as may be utilized in the currently preferred embodiment of the present invention.

Figure 6 illustrates a contents file layout for an individual digital work of the digital work of Figure 5 as may be utilized in the currently preferred embodiment of the present invention.

Figure 7 illustrates the components of a description block of the currently preferred embodiment of the present invention.

Figure 8 illustrates a description tree for the contents file layout of the digital work illustrated in Figure 5.

Figure 9 illustrates a portion of a description tree corresponding to the individual digital work illustrated in Figure 6.

Figure 10 illustrates a layout for the rights portion of a description block as may be utilized in the currently preferred embodiment of the present invention.

Figure 11 is a description tree wherein certain d-blocks have PRINT usage rights and is used to illustrate "strict" and "lenient" rules for resolving usage rights conflicts.

Figure 12 is a block diagram of the hardware components of a repository as are utilized in the currently preferred embodiment of the present invention.

Figure 13 is a block diagram of the functional (logical) components of a repository as are utilized in the currently preferred embodiment of the present invention.

Figure 14 is diagram illustrating the basic components of a usage right in the currently preferred embodiment of the present invention.

Figure 15 lists the usage rights grammar of the currently preferred embodiment of the present invention.

Figure 16 is a flowchart illustrating the steps of certificate delivery, hotlist checking and performance testing as performed in a registration transaction as may be performed in the currently preferred embodiment of the present invention.

Figure 17 is a flowchart illustrating the steps of session information exchange and clock synchronization as may be performed in the currently preferred embodiment of the present invention, after each repository in the registration transaction has successfully completed the steps described in Figure 16.

Figure 18 is a flowchart illustrating the basic flow for a usage transaction, including the common opening and closing step, as may be performed in the currently preferred embodiment of the present invention.

Figure 19 is a state diagram of server and client repositories in accordance with a transport protocol followed when moving a digital work from the server to the client repositories, as may be performed in the currently preferred embodiment of the present invention.

OVERVIEW

A system for controlling use and distribution of digital works is disclosed. The present invention is directed to supporting commercial transactions involving digital works.

Herein the terms "digital work", "work" and "content" refer to any work that has been reduced to a digital representation. This would include any audio, video, text, or multimedia work and any accompanying interpreter (e.g. software) that may be required for recreating the work. The term composite work refers to a digital work comprised of a collection of other digital works. The term "usage rights" or "rights" is a term which refers to rights granted to a recipient of a digital work. Generally, these rights define how a digital work can be used and if it can be further distributed. Each usage right may have one or more specified conditions which must be satisfied before the right may be exercised.

Figure 1 is a high level flowchart omitting various details but which demonstrates the basic operation of the present invention. Referring to Figure 1, a creator creates a digital work, step 101. The creator will then determine appropriate usage rights and fees, attach them to the digital work, and store them in Repository 1, step 102. The determination of appropriate usage rights and fees will depend on various economic factors. The digital work remains securely in Repository 1 until a request for access is received. The request for access begins with a session initiation by another repository. Here a Repository 2 initiates a session with Repository 1, step 103. As will be described in greater detail below, this session initiation includes steps which helps to insure that the respective repositories are trustworthy. As-

EPODOC: <EP 0715247A1 >

suming that a session can be established, Repository 2 may then request access to the Digital Work for a stated purpose, step 104. The purpose may be, for example, to print the digital work or to obtain a copy of the digital work. The purpose will correspond to a specific usage right. In any event, Repository 1 checks the usage rights associated with the digital work to determine if the access to the digital work may be granted, step 105. The check of the usage rights essentially involves a determination of whether a right associated with the access request has been attached to the digital work and if all conditions associated with the right are satisfied. If the access is denied, repository 1 terminates the session with an error message, step 106. If access is granted, repository 1 transmits the digital work to repository 2, step 107. Once the digital work has been transmitted to repository 2, repository 1 and 2 each generate billing information for the access which is transmitted to a credit server, step 108. Such double billing reporting is done to insure against attempts to circumvent the billing process.

Figure 2 illustrates the basic interactions between repository types in the present invention. As will become apparent from Figure 2, the various repository types will serve different functions. It is fundamental that repositories will share a core set of functionality which will enable secure and trusted communications. Referring to Figure 2, a repository 201 represents the general instance of a repository. The repository 201 has two modes of operation: a server mode and a requester mode. When in the server mode, the repository will be receiving and processing access requests to digital works. When in the requester mode, the repository will be initiating requests to access digital works. Repository 201 is general in the sense that its primary purpose is as an exchange medium for digital works. During the course of operation, the repository 201 may communicate with a plurality of other repositories, namely authorization repository 202, rendering repository 203 and master repository 204. Communication between repositories occurs utilizing a repository transaction protocol 205.

Communication with an authorization repository 202 may occur when a digital work being accessed has a condition requiring an authorization. Conceptually, an authorization is a digital certificate such that possession of the certificate is required to gain access to the digital work. An authorization is itself a digital work that can be moved between repositories and subjected to fees and usage rights conditions. An authorization may be required by both repositories involved in an access to a digital work.

Communication with a rendering repository 203 occurs in connection with the rendering of a digital work. As will be described in greater detail below, a rendering repository is coupled with a rendering device (e.g. a printer device) to comprise a rendering system.

Communication with a master repository 205 occurs in connection with obtaining an identification certificate. Identification certificates are the means by which a repository is identified as "trustworthy". The use of identification certificates is described below with respect to the registration transaction.

Figure 3 illustrates the repository 201 coupled to a credit server 301. The credit server 301 is a device which accumulates billing information for the repository 201. The credit server 301 communicates with repository 201 via billing transactions 302 to record billing transactions. Billing transactions are reported to a billing clearinghouse 303 by the credit server 301 on a periodic basis. The credit server 301 communicates to the billing clearinghouse 303 via clearinghouse transactions 304. The clearinghouse transactions 304 enable a secure and encrypted transmission of information to the billing clearinghouse 303.

RENDERING SYSTEMS

A rendering system is generally defined as a system comprising a repository and a rendering device which can render a digital work into its desired form. Examples of a rendering system may be a computer system, a digital audio system, or a printer. A rendering system has the same security features as a repository. The coupling of a rendering repository with the rendering device may occur in a manner suitable for the type of rendering device.

Figure 4a illustrates a printer as an example of a rendering system. Referring to Figure 4, printer system 401 has contained therein a printer repository 402 and a print device 403. It should be noted that the the dashed line defining printer system 401 defines a secure system boundary. Communications within the boundary are assumed to be secure. Depending on the security level, the boundary also represents a barrier intended to provide physical integrity. The printer repository 402 is an instantiation of the rendering repository 205 of Figure 2. The printer repository 402 will in some instances contain an ephemeral copy of a digital work which remains until it is printed out by the print engine 403. In other instances, the printer repository 402 may contain digital works such as fonts, which will remain and can be billed based on use. This design assures that all communication lines between printers and printing devices are encrypted, unless they are within a physically secure boundary. This design feature eliminates a potential "fault" point through which the digital work could be improperly obtained. The printer device 403 represents the printer components used to create the printed output.

Also illustrated in Figure 4a is the repository 404. The repository 404 is coupled to the printer repository 402. The repository 404 represents an external repository which contains digital works.

Figure 4b is an example of a computer system as a rendering system. A computer system may constitute a "multi-

BEDOCID: <EP 0715247A1 1 >

function" device since it may execute digital works (e.g. software programs) and display digital works (e.g. a digitized photograph). Logically, each rendering device can be viewed as having its own repository, although only one physical repository is needed. Referring to Figure 4b, a computer system 410 has contained therein a display/execution repository 411. The display/execution repository 411 is coupled to display device, 412 and execution device 413. The dashed box surrounding the computer system 410 represents a security boundary within which communications are assumed to be secure. The display/execution repository 411 is further coupled to a credit server 414 to report any fees to be billed for access to a digital work and a repository 415 for accessing digital works stored therein.

STRUCTURE OF DIGITAL WORKS

Usage rights are attached directly to digital works. Thus, it is important to understand the structure of a digital work. The structure of a digital work, in particular composite digital works, may be naturally organized into an acyclic structure such as a hierarchy. For example, a magazine has various articles and photographs which may have been created and are owned by different persons. Each of the articles and photographs may represent a node in a hierarchical structure. Consequently, controls, i.e. usage rights, may be placed on each node by the creator. By enabling control and fee billing to be associated with each node, a creator of a work can be assured that the rights and fees are not circumvented.

In the currently preferred embodiment, the file information for a digital work is divided into two files: a "contents" file and a "description tree" file. From the perspective of a repository, the "contents" file is a stream of addressable bytes whose format depends completely on the interpreter used to play, display or print the digital work. The description tree file makes it possible to examine the rights and fees for a work without reference to the content of the digital work. It should be noted that the term description tree as used herein refers to any type of acyclic structure used to represent the relationship between the various components of a digital work.

Figure 5 illustrates the layout of a contents file. Referring to Figure 5, a digital work is comprised of story A 510, advertisement 511, story B 512 and story C 513. It is assumed that the digital work is stored starting at a relative address of 0. Each of the parts of the digital work are stored linearly so that story A 510 is stored at approximately addresses 0-30,000, advertisement 511 at addresses 30,001-40,000, story B 512 at addresses 40,001-60,000 and story C 513 at addresses 60,001-85K. The detail of story A 510 is illustrated in Figure 6. Referring to Figure 6, the story A 510 is further broken down to show text 614 stored at address 0-1500, soldier photo 615 at addresses 1501-10,000, graphics 616 stored at addresses 10,001-25,000 and sidebar 617 stored address 25,001-30,000. Note that the data in the contents file may be compressed (for saving storage) or encrypted (for security).

From Figures 5 and 6 it is readily observed that a digital work can be represented by its component parts as a hierarchy. The description tree for a digital work is comprised of a set of related descriptor blocks (d-blocks). The contents of each d-block is described with respect to Figure 7. Referring to Figure 7, a d-block 700 includes an identifier 701 which is a unique identifier for the work in the repository, a starting address 702 providing the start address of the first byte of the work, a length 703 giving the number of bytes in the work, a rights portion 704 wherein the granted usage rights and their status data are maintained, a parent pointer 705 for pointing to a parent d-block and child pointers 706 for pointing to the child d-blocks. In the currently preferred embodiment, the identifier 701 has two parts. The first part is a unique number assigned to the repository upon manufacture. The second part is a unique number assigned to the work upon creation. The rights portion 704 will contain a data structure, such as a look-up table, wherein the various information associated with a right is maintained. The information required by the respective usage rights is described in more detail below. D-blocks form a strict hierarchy. The top d-block of a work has no parent; all other d-blocks have one parent. The relationship of usage rights between parent and child d-blocks and how conflicts are resolved is described below.

A special type of d-block is a "shell" d-block. A shell d-block adds no new content beyond the content of its parts. A shell d-block is used to add rights and fee information, typically by distributors of digital works.

Figure 8 illustrates a description tree for the digital work of Figure 5. Referring to Figure 8, a top d-block 820 for the digital work points to the various stories and advertisements contained therein. Here, the top d-block 820 points to d-block 821 (representing story A 510), d-block 822 (representing the advertisement 511), d-block 823 (representing story B 512) and and d-block 824 (representing story C 513).

The portion of the description tree for Story A 510 is illustrated in Figure 9. D-block 925 represents text 614, d-block 926 represents photo 615, d-block 927 represents graphics 616 by and d-block 928 represents sidebar 617.

The rights portion 704 of a descriptor block is further illustrated in Figure 10. Figure 10 illustrates a structure which is repeated in the rights portion 704 for each right. Referring to Figure 10, each right will have a right code field 1050 and status information field 1052. The right code field 1050 will contain a unique code assigned to a right. The status information field 1052 will contain information relating to the state of a right and the digital work. Such information is indicated below in Table 1. The rights as stored in the rights portion 704 may typically be in numerical order based on the right code.

INSDOCID <EP 0715247A1_I_>

TABLE 1

DIGITAL WORK STATE INFORMATION		
Property	**Value**	**Use**
Copies-in-Use	Number	A counter of the number of copies of a work that are in use. Incremented when another copy is used, decremented when use is completed.
Loan-Period	Time-Units	Indicator of the maximum number of time-units that a document can be loaned out
Loaner-Copy	Boolean	Indicator that the current work is a loaned out copy of an authorized digital work.
Remaining-Time	Time-Units	Indicator of the remaining time of use on a metered document right.
Document-Descr	String	A string containing various identifying information about a document. The exact format of this is not specified, but it can include information such as a publisher name, author name, ISBN number, and so on.
Revenue-Owner	RO-Descr	A handle identifying a revenue owner for a digital work. This is used for reporting usage fees.
Publication-Date	Date-Descr	The date that the digital work was published.
History-list	History-Rec	A list of events recording the repositories and dates for operations that copy, transfer, backup, or restore a digital work.

The approach for representing digital works by separating description data from content assumes that parts of a file are contiguous but takes no position on the actual representation of content. In particular, it is neutral to the question of whether content representation may take an object oriented approach. It would be natural to represent content as objects. In principle, it may be convenient to have content objects that include the billing structure and rights information that is represented in the d-blocks. Such variations in the design of the representation are possible and are viable alternatives but may introduce processing overhead, e.g. the interpretation of the objects.

Digital works are stored in a repository as part of a hierarchical file system. Folders (also termed directories and sub-directories) contain the digital works as well as other folders. Digital works and folders in a folder are ordered in alphabetical order. The digital works are typed to reflect how the files are used. Usage rights can be attached to folders so that the folder itself is treated as a digital work. Access to the folder would then be handled in the same fashion as any other digital work. As will be described in more detail below, the contents of the folder are subject to their own rights. Moreover, file management rights may be attached to the folder which define how folder contents can be managed.

ATTACHING USAGE RIGHTS TO A DIGITAL WORK

It is fundamental to the present invention that the usage rights are treated as part of the digital work. As the digital work is distributed, the scope of the granted usage rights will remain the same or may be narrowed. For example, when a digital work is transferred from a document server to a repository, the usage rights may include the right to loan a copy for a predetermined period of time (called the original rights). When the repository loans out a copy of the digital work, the usage rights in the loaner copy (called the next set of rights) could be set to prohibit any further rights to loan out the copy. The basic idea is that one cannot grant more rights than they have.

The attachment of usage rights into a digital work may occur in a variety of ways. If the usage rights will be the same for an entire digital work, they could be attached when the digital work is processed for deposit in the digital work server. In the case of a digital work having different usage rights for the various components, this can be done as the digital work is being created. An authoring tool or digital work assembling tool could be utilized which provides for an automated process of attaching the usage rights.

As will be described below, when a digital work is copied, transferred or loaned, a "next set of rights" can be specified. The "next set of rights" will be attached to the digital work as it is transported.

Resolving Conflicting Rights

Because each part of a digital work may have its own usage rights, there will be instances where the rights of a "contained part" are different from its parent or container part. As a result, conflict rules must be established to dictate when and how a right may be exercised. The hierarchical structure of a digital work facilitates the enforcement of such

XEROX0D <EP 0715247A1 >

rules. A "strict" rule would be as follows: a right for a part in a digital work is sanctioned if and only if it is sanctioned for the part, for ancestor d-blocks containing the part and for all descendent d-blocks. By sanctioned, it is meant that (1) each of the respective parts must have the right, and (2) any conditions for exercising the right are satisfied.

It also possible to implement the present invention using a more lenient rule. In the more lenient rule, access to the part may be enabled to the descendent parts which have the right, but access is denied to the descendents which do not.

An example of applying both the strict rule and lenient is illustrated with reference to Figure 11. Referring to Figure 11, a root d-block 1101 has child d-blocks 1102-1105. In this case, root d-block represents a magazine, and each of the child d-blocks 1102-1105 represent articles in the magazine. Suppose that a request is made to PRINT the digital work represented by root d-block 1101 wherein the strict rule is followed. The rights for the root d-block 1101 and child d-blocks 1102-1105 are then examined. Root d-block 1101 and child d-blocks 1102 and 1105 have been granted PRINT rights. Child d-block 1103 has not been granted PRINT rights and child d-block 1104 has PRINT rights conditioned on payment of a usage fee.

Under the strict rule the PRINT right cannot be exercised because the child d-block does not have the PRINT right. Under the lenient rule, the result would be different. The digital works represented by child d-blocks 1102 and 1105 could be printed and the digital work represented by d-block 1104 could be printed so long as the usage fee is paid. Only the digital work represented by d-block 1103 could not be printed. This same result would be accomplished under the strict rule if the requests were directed to each of the individual digital works.

The present invention supports various combinations of allowing and disallowing access. Moreover, as will be described below, the usage rights grammar permits the owner of a digital work to specify if constraints may be imposed on the work by a container part. The manner in which digital works may be sanctioned because of usage rights conflicts would be implementation specific and would depend on the nature of the digital works

REPOSITORIES

In the description of Figure 2, it was indicated that repositories come in various forms. All repositories provide a core set of services for the transmission of digital works. The manner in which digital works are exchanged is the basis for all transaction between repositories. The various repository types differ in the ultimate functions that they perform. Repositories may be devices themselves, or they may be incorporated into other systems. An example is the rendering repository 203 of Figure 2.

A repository will have associated with it a repository identifier. Typically, the repository identifier would be a unique number assigned to the repository at the time of manufacture. Each repository will also be classified as being in a particular security class. Certain communications and transactions may be conditioned on a repository being in a particular security class. The various security classes are described in greater detail below.

As a prerequisite to operation, a repository will require possession of an identification certificate. Identification certificates are encrypted to prevent forgery and are issued by a Master repository. A master repository plays the role of an authorization agent to enable repositories to receive digital works. Identification certificates must be updated on a periodic basis. Identification certificates are described in greater detail below with respect to the registration transaction.

A repository has both a hardware and functional embodiment. The functional embodiment is typically software executing on the hardware embodiment. Alternatively, the functional embodiment may be embedded in the hardware embodiment such as an Application Specific Integrated Circuit (ASIC) chip.

The hardware embodiment of a repository will be enclosed in a secure housing which if compromised, may cause the repository to be disabled. The basic components of the hardware embodiment of a repository are described with reference to Figure 12. Referring to Figure 12, a repository is comprised of a processing means 1200, storage system 1207, clock 1205 and external interface 1206. The processing means 1200 is comprised of a processor element 1201 and processor memory 1202. The processing means 1201 provides controller repository transaction and usage rights transaction functions for the repository. Various functions in the operation of the repository such as decryption and/or decompression of digital works and transaction messages are also performed by the processing means 1200. The processor element 1201 may be a microprocessor or other suitable computing component. The processor memory 1202 would typically be further comprised of Read Only Memories (ROM) and Random Access Memories (RAM). Such memories would contain the software instructions utilized by the processor element 1201 in performing the functions of the repository.

The storage system 1207 is further comprised of descriptor storage 1203 and content storage 1204. The description tree storage 1203 will store the description tree for the digital work and the content storage will store the associated content. The description tree storage 1203 and content storage 1204 need not be of the same type of storage medium, nor are they necessarily on the same physical device. So for example, the descriptor storage 1203 may be stored on a solid state storage (for rapid retrieval of the description tree information), while the content storage 1204 may be on

ISDOCID <EP_____0715247A1_I_>

a high capacity storage such as an optical disk.

The clock 1205 is used to time-stamp various time based conditions for usage rights or for metering usage fees which may be associated with the digital works. The clock 1205 will have an uninterruptable power supply, e.g. a battery, in order to maintain the integrity of the time-stamps. The external interface means 1206 provides for the signal connection to other repositories and to a credit server. The external interface means 1206 provides for the exchange of signals via such standard interfaces such as RS-232 or Personal Computer Manufacturers Card Industry Association (PCMCIA) standards, or FDDI. The external interface means 1206 may also provide network connectivity.

The functional embodiment of a repository is described with reference to Figure 13. Referring to Figure 13, the functional embodiment is comprised of an operating system 1301, core repository services 1302, usage transaction handlers 1303, repository specific functions, 1304 and a user interface 1305. The operating system 1301 is specific to the repository and would typically depend on the type of processor being used. The operating system 1301 would also provide the basic services for controlling and interfacing between the basic components of the repository.

The core repository services 1302 comprise a set of functions required by each and every repository. The core repository services 1302 include the session initiation transactions which are defined in greater detail below. This set of services also includes a generic ticket agent which is used to "punch" a digital ticket and a generic authorization server for processing authorization specifications. Digital tickets and authorizations are specific mechanisms for controlling the distribution and use of digital works and are described in more detail below. Note that coupled to the core repository services are a plurality of identification certificates 1306. The identification certificates 1306 are required to enable the use of the repository.

The usage transactions handlers 1303 comprise functionality for processing access requests to digital works and for billing fees based on access. The usage transactions supported will be different for each repository type. For example, it may not be necessary for some repositories to handle access requests for digital works.

The repository specific functionality 1304 comprises functionality that is unique to a repository. For example, the master repository has special functionality for issuing digital certificates and maintaining encryption keys. The repository specific functionality 1304 would include the user interface implementation for the repository.

Repository Security Classes

For some digital works the losses caused by any individual instance of unauthorized copying is insignificant and the chief economic concern lies in assuring the convenience of access and low-overhead billing. In such cases, simple and inexpensive handheld repositories and network-based workstations may be suitable repositories, even though the measures and guarantees of security are modest.

At the other extreme, some digital works such as a digital copy of a first run movie or a bearer bond or stock certificate would be of very high value so that it is prudent to employ caution and fairly elaborate security measures to ensure that they are not copied or forged. A repository suitable for holding such a digital work could have elaborate measures for ensuring physical integrity and for verifying authorization before use.

By arranging a universal protocol, all kinds of repositories can communicate with each other in principle. However, creators of some works will want to specify that their works will only be transferred to repositories whose level of security is high enough. For this reason, document repositories have a ranking system for classes and levels of security. The security classes in the currently preferred embodiment are described in Table 2.

TABLE 2

REPOSITORY SECURITY LEVELS	
Level	Description of Security
0	Open system. Document transmission is unencrypted. No digital certificate is required for identification. The security of the system depends mostly on user honesty, since only modest knowledge may be needed to circumvent the security measures. The repository has no provisions for preventing unauthorized programs from running and accessing or copying files. The system does not prevent the use of removable storage and does not encrypt stored files.
1	Minimal security. Like the previous class except that stored files are minimally encrypted, including ones on removable storage.
2	Basic security. Like the previous class except that special tools and knowledge are required to compromise the programming, the contents of the repository, or the state of the clock. All digital communications are encrypted. A digital certificate is provided as identification. Medium level encryption is used. Repository identification number is unforgeable.

3DOCID: <EP_____0715247A1_1_>

TABLE 2 (continued)

REPOSITORY SECURITY LEVELS	
Level	Description of Security
3	General security. Like the previous class plus the requirement of special tools are needed to compromise the physical integrity of the repository and that modest encryption is used on all transmissions. Password protection is required to use the local user interface. The digital clock system cannot be reset without authorization. No works would be stored on removable storage. When executing works as programs, it runs them in their own address space and does not give them direct access to any file storage or other memory containing system code or works. They can access works only through the transmission transaction protocol.
4	Like the previous class except that high level encryption is used on all communications. Sensors are used to record attempts at physical and electronic tampering. After such tampering, the repository will not perform other transactions until it has reported such tampering to a designated server.
5	Like the previous class except that if the physical or digital attempts at tampering exceed some preset thresholds that threaten the physical integrity of the repository or the integrity of digital and cryptographic barriers, then the repository will save only document description records of history but will erase or destroy any digital identifiers that could be misused if released to an unscrupulous party. It also modifies any certificates of authenticity to indicate that the physical system has been compromised. It also erases the contents of designated documents.
6	Like the previous class except that the repository will attempt wireless communication to report tampering and will employ noisy alarms.
10	This would correspond to a very high level of security. This server would maintain constant communications to remote security systems reporting transactions, sensor readings, and attempts to circumvent security.

The characterization of security levels described in Table 2 is not intended to be fixed. More important is the idea of having different security levels for different repositories. It is anticipated that new security classes and requirements will evolve according to social situations and changes in technology.

Repository User Interface

A user interface is broadly defined as the mechanism by which a user interacts with a repository in order to invoke transactions to gain access to a digital work, or exercise usage rights. As described above, a repository may be embodied in various forms. The user interface for a repository will differ depending on the particular embodiment. The user interface may be a graphical user interface having icons representing the digital works and the various transactions that may be performed. The user interface may be a generated dialog in which a user is prompted for information

The user interface itself need not be part of the repository. As a repository may be embedded in some other device, the user interface may merely be a part of the device in which the repository is embedded. For example, the repository could be embedded in a "card" that is inserted into an available slot in a computer system. The user interface may be a combination of a display, keyboard, cursor control device and software executing on the computer system.

At a minimum, the user interface must permit a user to input information such as access requests and alpha numeric data and provide feedback as to transaction status. The user interface will then cause the repository to initiate the suitable transactions to service the request. Other facets of a particular user interface will depend on the functionality that a repository will provide.

CREDIT SERVERS

In the present invention, fees may be associated with the exercise of a right. The requirement for payment of fees is described with each version of a usage right in the usage rights language. The recording and reporting of such fees is performed by the credit server. One of the capabilities enabled by associating fees with rights is the possibility of supporting a wide range of charging models. The simplest model, used by conventional software, is that there is a single fee at the time of purchase, after which the purchaser obtains unlimited rights to use the work as often and for as long as he or she wants. Alternative models, include metered use and variable fees. A single work can have different fees for different uses. For example, viewing a photograph on a display could have different fees than making a hardcopy

or including it in a newly created work. A key to these alternative charging models is to have a low overhead means of establishing fees and accounting for credit on these transactions.

A credit server is a computational system that reliably authorizes and records these transactions so that fees are billed and paid. The credit server reports tees to a billing clearinghouse. The billing clearinghouse manages the financial transactions as they occur. As a result, bills may be generated and accounts reconciled. Preferably, the credit server would store the fee transactions and periodically communicate via a network with the billing clearinghouse for reconciliation. In such an embodiment, communications with the billing clearinghouse would be encrypted for integrity and security reasons. In another embodiment, the credit server acts as a "debit card" where transactions occur in "real-time" against a user account.

A credit server is comprised of memory, a processing means, a clock, and interface means for coupling to a repository and a financial institution (e.g. a modem). The credit server will also need to have security and authentication functionality. These elements are essentially the same elements as those of a repository. Thus, a single device can be both a repository and a credit server, provided that it has the appropriate processing elements for carrying out the corresponding functions and protocols. Typically, however, a credit server would be a card-sized system in the possession of the owner of the credit. The credit server is coupled to a repository and would interact via financial transactions as described below. Interactions with a financial institution may occur via protocols established by the financial institutions themselves.

In the currently preferred embodiment credit servers associated with both the server and the repository report the financial transaction to the billing clearinghouse. For example, when a digital work is copied by one repository to another for a fee, credit servers coupled to each of the repositories will report the transaction to the billing clearinghouse. This is desirable in that it insures that a transaction will be accounted for in the event of some break in the communication between a credit server and the billing clearinghouse. However some implementations may embody only a single credit server reporting the transaction to minimize transaction processing at the risk of losing some transactions.

USAGE RIGHTS LANGUAGE

The present invention uses statements in a high level "usage rights language" to define rights associated with digital works and their parts. Usage rights statements are interpreted by repositories and are used to determine what transactions can be successfully carried out for a digital work and also to determine parameters for those transactions. For example, sentences in the language determine whether a given digital work can be copied, when and how it can be used, and what fees (if any) are to be charged for that use. Once the usage rights statements are generated, they are encoded in a suitable form for accessing during the processing of transactions.

Defining usage rights in terms of a language in combination with the hierarchical representation of a digital work enables the support of a wide variety of distribution and fee schemes. An example is the ability to attach multiple versions of a right to a work. So a creator may attach a PRINT right to make 5 copies for $10.00 and a PRINT right to make unlimited copies for $100.00. A purchaser may then choose which option best fits his needs. Another example is that rights and fees are additive. So in the case of a composite work, the rights and fees of each of the components works is used in determining the rights and fees for the work as a whole.

The basic contents of a right are illustrated in Figure 14. Referring to Figure 14, a right 1450 has a transactional component 1451 and a specifications component 1452. A right 1450 has a label (e.g. COPY or PRINT) which indicates the use or distribution privileges that are embodied by the right. The transactional component 1451 corresponds to a particular way in which a digital work may be used or distributed. The transactional component 1451 is typically embodied in software instructions in a repository which implement the use or distribution privileges for the right. The specifications components 1452 are used to specify conditions which must be satisfied prior to the right being exercised or to designate various transaction related parameters. In the currently preferred embodiment, these specifications include copy count 1453, Fees and Incentives 1454, Time 1455, Access and Security 1456 and Control 1457. Each of these specifications will be described in greater detail below with respect to the language grammar elements.

The usage rights language is based on the grammar described below. A grammar is a convenient means for defining valid sequence of symbols for a language. In describing the grammar the notation "[a|b|c]" is used to indicate distinct choices among alternatives. In this example, a sentence can have either an "a", "b" or "c". It must include exactly one of them. The braces {} are used to indicate optional items. Note that brackets, bars and braces are used to describe the language of usage rights sentences but do not appear in actual sentences in the language.

In contrast, parentheses are part of the usage rights language. Parentheses are used to group items together in lists. The notation (x*) is used to indicate a variable length list, that is, a list containing one or more items of type x. The notation (x)* is used to indicate a variable number of lists containing x.

Keywords in the grammar are words followed by colons. Keywords are a common and very special case in the language. They are often used to indicate a single value, typically an identifier. In many cases, the keyword and the parameter are entirely optional. When a keyword is given, it often takes a single identifier as its value. In some cases,

DOCID: <EP_____0715247A1_I_>

the keyword takes a list of identifiers.

In the usage rights language, time is specified in an hours:minutes:seconds (or hh:mm:ss) representation. Time zone indicators, e.g. PDT for Pacific Daylight Time, may also be specified. Dates are represented as year/ month/day (or YYYY/MMM/DD). Note that these time and date representations may specify moments in time or units of time. Money units are specified in terms of dollars.

Finally, in the usage rights language, various "things" will need to interact with each other. For example, an instance of a usage right may specify a bank account, a digital ticket, etc.. Such things need to be identified and are specified herein using the suffix "-ID."

The Usage Rights Grammar is listed in its entirety in Figure 15 and is described below.

Grammar element 1501 **"Digital Work Rights: = (Rights*)"** define the digital work rights as a set of rights. The set of rights attached to a digital work define how that digital work may be transferred, used, performed or played. A set of rights will attach to the entire digital work and in the case of compound digital works, each of the components of the digital work. The usage rights of components of a digital may be different.

Grammar element 1502 **"Right : = (Right-Code {Copy-Count} {Control-Spec} {Time-Spec}{Access-spec} {Fee-Spec})"** enumerates the content of a right. Each usage right must specify a right code. Each right may also optionally specify conditions which must be satisfied before the right can be exercised. These conditions are copy count, control, time, access and fee conditions. In the currently preferred embodiment, for the optional elements, the following defaults apply: copy count equals 1, no time limit on the use of the right, no access tests or a security level required to use the right and no fee is required. These conditions will each be described in greater detail below.

It is important to note that a digital work may have multiple versions of a right, each having the same right code. The multiple version would provide alternative conditions and fees for accessing the digital work.

Grammar element 1503 **"Right-Code : = Render-Code I Transport-Code I File-Management-Code I Derivative-Works- Code Configuration-Code"** distinguishes each of the specific rights into a particular right type (although each right is identified by distinct right codes). In this way, the grammar provides a catalog of possible rights that can be associated with parts of digital works. In the following, rights are divided into categories for convenience in describing them.

Grammar element 1504 "Render-Code : = [Play : {Player: Player-ID}IPrint:{Printer: Printer-ID}]" lists a category of rights all involving the making of ephemeral, transitory, or non-digital copies of the digital work. After use the copies are erased.

- Play — A process of rendering or performing a digital work on some processor. This includes such things as playing digital movies, playing digital music, playing a video game, running a computer program, or displaying a document on a display.
- Print — To render the work in a medium that is not further protected by usage rights, such as printing on paper.

Grammar element 1505 **"Transport-Code : = [CopyITransferILoan {Remaining-Rights: Next-Set-of-Rights}] {[Next-Copy.Rights: Next-Set of Rights)]"** lists a category of rights involving the making of persistent, usable copies of the digital work on other repositories. The optional Next-Copy-Rights determine the rights on the work after it is transported. If this is not specified, then the rights on the transported copy are the same as on the original. The optional Remaining-Rights specify the rights that remain with a digital work when it is loaned out. If this is not specified, then the default is that no rights can be exercised when it is loaned out.

- Copy — Make a new copy of a work
- Transfer — Moving a work from one repository to another.
- Loan — Temporarily loaning a copy to another repository for a specified period of time

Grammar element 1506 **"File-Management-Code : = Backup {Back-Up-Copy-Rights: Next-Set -of Rights}I Restore I Delete I Folder I Directory {Name:Hide-Local I Hide - Remote}{Parts:Hide-Local I Hide-Remote}"** lists a category of rights involving operations for file management, such as the making of backup copies to protect the copy owner against catastrophic equipment failure.

Many software licenses and also copyright law give a copy owner the right to make backup copies to protect against catastrophic failure of equipment. However, the making of uncontrolled backup copies is inherently at odds with the ability to control usage, since an uncontrolled backup copy can be kept and then restored even after the authorized copy was sold.

The File management rights enable the making and restoring of backup copies in a way that respects usage rights, honoring the requirements of both the copy owner and the rights grantor and revenue owner. Backup copies of work descriptions (including usage rights and fee data) can be sent under appropriate protocol and usage rights control to other document repositories of sufficiently high security. Further rights permit organization of digital works into folders

SDOCID: <EP___ ___ 0715247A1_I_>

which themselves are treated as digital works and whose contents may be "hidden" from a party seeking to determine the contents of a repository.

- Backup To make a backup copy of a digital work as protection against media failure.
- Restore To restore a backup copy of a digital work.
- Delete To delete or erase a copy of a digital work.
- Folder To create and name folders, and to move files and folders between folders.
- Directory To hide a folder or its contents.

Grammar element 1507 **"Derivative-Works-Code: [Extract | Embed | Edit {Process: Process-ID}] (Next-Copy-Rights : Next-Set-of Rights)"** lists a category of rights involving the use of a digital work to create new works.

- Extract To remove a portion of a work, for the purposes of creating a new work.
- Embed To include a work in an existing work.
- Edit To alter a digital work by copying, selecting and modifying portions of an existing digital work.

Grammar element 1508 **"Configuration-Code: = Install | Uninstall"** lists a category of rights for installing and uninstalling software on a repository (typically a rendering repository.) This would typically occur for the installation of a new type of player within the rendering repository.

- Install: To install new software on a repository.
- Uninstall: To remove existing software from a repository.

Grammar element 1509 **"Next-Set-of-Rights : = {(Add : Set-Of-Rights)} {(Delete: Set-Of-Rights)} {(Replace: Set-Of-Rights)} {(Keep: Set-Of-Rights)}"** defines how rights are carried forward for a copy of a digital work. If the Next-Copy-Rights is not specified, the rights for the next copy are the same as those of the current copy. Otherwise, the set of rights for the next copy can be specified. Versions of rights after Add: are added to the current set of rights. Rights after Delete: are deleted from the current set of rights. If only right codes are listed after Delete:, then all versions of rights with those codes are deleted. Versions of rights after Replace: subsume all versions of rights of the same type in the current set of rights.

If Remaining-Rights is not specified, then there are no rights for the original after all Loan copies are loaned out. If Remaining-Rights is specified, then the Keep: token can be used to simplify the expression of what rights to keep behind. A list of right codes following keep means that all of the versions of those listed rights are kept in the remaining copy. This specification can be overridden by subsequent Delete: or Replace: specifications.

Copy Count Specification

For various transactions, it may be desirable to provide some limit as to the number of "copies" of the work which may be exercised simultaneously for the right. For example, it may be desirable to limit the number of copies of a digital work that may be loaned out at a time or viewed at a time.

Grammar element 1510 **"Copy-Count : = (Copies: positive-integer | 0 | unlimited)"** provides a condition which defines the number of "copies" of a work subject to the right. A copy count can be 0, a fixed number, or unlimited. The copy-count is associated with each right, as opposed to there being just a single copy-count for the digital work. The Copy-Count for a right is decremented each time that a right is exercised. When the Copy-Count equals zero, the right can no longer be exercised. If the Copy-Count is not specified, the default is one.

Control Specification

Rights and fees depend in general on rights granted by the creator as well as further restrictions imposed by later distributors. Control specifications deal with interactions between the creators and their distributors governing the imposition of further restrictions and fees. For example, a distributor of a digital work may not want an end consumer of a digital work to add fees or otherwise profit by commercially exploiting the purchased digital work.

Grammar element 1511 **"Control-Spec : = (Control: {Restrictable | Unrestrictable} {Unchargeable | Chargeable})"** provides a condition to specify the effect of usage rights and fees of parents on the exercise of the right. A digital work is restrictable if higher level d-blocks can impose further restrictions (time specifications and access specifications) on the right. It is unrestrictable if no further restrictions can be imposed. The default setting is restrictable. A right is unchargeable if no more fees can be imposed on the use of the right. It is chargeable if more fees can be imposed. The default is chargeable.

EP 0 715 247 A1

Time Specification

It is often desirable to assign a start date or specify some duration as to when a right may be exercised. Grammar element 1512 **"Time-Spec : = ({Fixed-Interval I Sliding-Interval I Meter-Time} Until: Expiration-Date)"** provides for specification of time conditions on the exercise of a right. Rights may be granted for a specified time. Different kinds of time specifications are appropriate for different kinds of rights. Some rights may be exercised during a fixed and predetermined duration. Some rights may be exercised for an interval that starts the first time that the right is invoked by some transaction. Some rights may be exercised or are charged according to some kind of metered time, which may be split into separate intervals. For example, a right to view a picture for an hour might be split into six ten minute viewings or four fifteen minute viewings or twenty three minute viewings.

The terms "time" and "date" are used synonymously to refer to a moment in time. There are several kinds of time specifications. Each specification represents some limitation on the times over which the usage right applies. The Expiration-Date specifies the moment at which the usage right ends. For example, if the Expiration-Date is "Jan 1, 1996," then the right ends at the first moment of 1995. If the Expiration-Date is specified as "forever", then the rights are interpreted as continuing without end. It only an expiration date is given, then the right can be exercised as often as desired until the expiration date.

Grammar element 1513 **"Fixed-Interval : = From: Start-Time"** is used to define a predetermined interval that runs from the start time to the expiration date.

Grammar element 1514 **"Sliding-Interval : := Interval: Use-Duration"** is used to define an indeterminate (or "open") start time. It sets limits on a continuous period of time over which the contents are accessible. The period starts on the first access and ends after the duration has passed or the expiration date is reached, whichever comes first. For example, if the right gives 10 hours of continuous access, the use-duration would begin when the first access was made and end 10 hours later.

Grammar element 1515 **"Meter-Time: = Time-Remaining: Remaining-Use"** is used to define a "meter time," that is, a measure of the time that the right is actually exercised. It differs from the Sliding-Interval specification in that the time that the digital work is in use need not be continuous. For example, if the rights guarantee three days of access, those days could be spread out over a month. With this specification, the rights can be exercised until the meter time is exhausted or the expiration date is reached, whichever comes first.

Remaining-Use: = Time-Unit
Start-Time: = Time-Unit
Use-Duration: = Time-Unit

All of the time specifications include time-unit specifications in their ultimate instantiation.

Security Class and Authorization Specification

The present invention provides for various security mechanisms to be introduced into a distribution or use scheme. Grammar element 1516 **"Access-Spec:=({SC: Security-Class}{Authorization: Authorization.ID*}{Other-Authorization: Authorization-ID*}{Ticket: Ticket-ID})"** provides a means for restricting access and transmission. Access specifications can specify a required security class for a repository to exercise a right or a required authorization test that must be satisfied.

The keyword **"SC:"** is used to specify a minimum security level for the repositories involved in the access. If "SC: " is not specified, the lowest security level is acceptable.

The optional **"Authorization:"** keyword is used to specify required authorizations on the same repository as the work. The optional **"Other-Authorization:"** keyword is used to specify required authorizations on the other repository in the transaction.

The optional **"Ticket:"** keyword specifies the identity of a ticket required for the transaction. A transaction involving digital tickets must locate an appropriate digital ticket agent who can "punch" or otherwise validate the ticket before the transaction can proceed. Tickets are described in greater detail below.

In a transaction involving a repository and a document server, some usage rights may require that the repository have a particular authorization, that the server have some authorization, or that both repositories have (possibly different) authorizations. Authorizations themselves are digital works (hereinafter referred to as an authorization object) that can be moved between repositories in the same manner as other digital works. Their copying and transferring is subject to the same rights and fees as other digital works. A repository is said to have an authorization if that authorization object is contained within the repository.

In some cases, an authorization may be required from a source other than the document server and repository. An authorization object referenced by an Authorization-ID can contain digital address information to be used to set up

a communications link between a repository and the authorization source. These are analogous to phone numbers. For such access tests, the communication would need to be established and authorization obtained before the right could be exercised.

For one-time usage rights, a variant on this scheme is to have a digital ticket. A ticket is presented to a digital ticket agent, whose type is specified on the ticket. In the simplest case, a certified generic ticket agent, available on all repositories, is available to "punch" the ticket. In other cases, the ticket may contain addressing information for locating a "special" ticket agent. Once a ticket has been punched, it cannot be used again for the same kind of transaction (unless it is unpunched or refreshed in the manner described below.) Punching includes marking the ticket with a timestamp of the date and time it was used. Tickets are digital works and can be copied or transferred between repositories according to their usage rights.

In the currently preferred embodiment, a "punched" ticket becomes "unpunched" or "refreshed" when it is copied or extracted. The Copy and Extract operations save the date and time as a property of the digital ticket. When a ticket agent is given a ticket, it can simply check whether the digital copy was made after the last time that it was punched. Of course, the digital ticket must have the copy or extract usage rights attached thereto.

The capability to unpunch a ticket is important in the following cases:

- A digital work is circulated at low cost with a limitation that it can be used only once.
- A digital work is circulated with a ticket that can be used once to give discounts on purchases of other works
- A digital work is circulated with a ticket (included in the purchase price and possibly embedded in the work) that can be used for a future upgrade.

In each of these cases, if a paid copy is made of the digital work (including the ticket) the new owner would expect to get a fresh (unpunched) ticket, whether the copy seller has used the work or not. In contrast, loaning a work or simply transferring it to another repository should not revitalize the ticket.

Usage Fees and Incentives Specification

The billing for use of a digital work is fundamental to a commercial distribution system. Grammar Element 1517 **"Fee-Spec: = {scheduled-Discount) Regular-Fee-Spec I Scheduled-Fee-Spec I Markup-Spec"** provides a range of options for billing for the use of digital works.

A key feature of this approach is the development of low-overhead billing for transactions in potentially small amounts. Thus, it becomes feasible to collect fees of only a few cents each for thousands of transactions.

The grammar differentiates between uses where the charge is per use from those where it is metered by the time unit. Transactions can support fees that the user pays for using a digital work as well as incentives paid by the right grantor to users to induce them to use or distribute the digital work.

The optional scheduled discount refers to the rest of the fee specification--discounting it by a percentage over time. If it is not specified, then there is no scheduled discount. Regular fee specifications are constant over time. Scheduled fee specifications give a schedule of dates over which the fee specifications change. Markup specifications are used in d-blocks for adding a percentage to the fees already being charged.

Grammar Element 1518 **"Scheduled-Discount:= (Scheduled-Discount: (Time-Spec Percentage)*)"** A Scheduled-Discount is a essentially a scheduled modifier of any other fee specification for this version of the right of the digital work. (It does not refer to children or parent digital works or to other versions of rights.). It is a list of pairs of times and percentages. The most recent time in the list that has not yet passed at the time of the transaction is the one in effect. The percentage gives the discount percentage. For example, the number 10 refers to a 10% discount.

Grammar Element 1519 **"Regular-Fee-Spec : = ({Fee: I Incentive: } [Per-Use-Spec I Metered-Rate-Spec I Best-Price-Spec I Call-For-Price-Spec] (Min: Money-Unit Per: Time-Spec}{Max: Money-Unit Per: Time-Spec) To: Account-ID)"** provides for several kinds of fee specifications.

Fees are paid by the copy-owner/user to the revenue-owner if **Fee**: is specified. Incentives are paid by the revenue-owner to the user if **Incentive**: is specified. If the **Min**: specification is given, then there is a minimum fee to be charged per time-spec unit for its use. If the **Max**: specification is given, then there is a maximum fee to be charged per time-spec for its use. When **Fee**: is specified, **Account-ID** identifies the account to which the fee is to be paid. When **Incentive**: is specified, **Account-ID** identifies the account from which the fee is to be paid.

Grammar element 1520 **"Per-Use-Spec: = Per-Use: Money-unit"** defines a simple fee to be paid every time the right is exercised, regardless of how much time the transaction takes.

Grammar element 1521 **"Metered-Rate-Spec : = Metered: Money-Unit Per: Time-Spec"** defines a metered-rate fee paid according to how long the right is exercised. Thus, the time it takes to complete the transaction determines the fee.

Grammar element 1522 **"Best-Price-Spec : = Best-Price: Money-unit Max: Money-unit"** is used to specify a

EDOCID: <EP 0715247A1 I >

best-price that is determined when the account is settled. This specification is to accommodate special deals, rebates, and pricing that depends on information that is not available to the repository. All fee specifications can be combined with tickets or authorizations that could indicate that the consumer is a wholesaler or that he is a preferred customer, or that the seller be authorized in some way. The amount of money in the Max: field is the maximum amount that the use will cost. This is the amount that is tentatively debited from the credit server. However, when the transaction is ultimately reconciled, any excess amount will be returned to the consumer in a separate transaction.

Grammar element 1523 **"Call-For-Price-Spec: = Call-For-Price "** is similar to a **"Best-Price-Spec"** in that it is intended to accommodate cases where prices are dynamic. **A Call-For-Price Spec** requires a communication with a dealer to determine the price. This option cannot be exercised if the repository cannot communicate with a dealer at the time that the right is exercised. It is based on a secure transaction whereby the dealer names a price to exercise the right and passes along a deal certificate which is referenced or included in the billing process.

Grammar element 1524 **"Scheduled-Fee-Spec: = {Schedule: (Time-Spec Regular-Fee-Spec)*)"** is used to provide a schedule of dates over which the fee specifications change. The fee specification with the most recent date not in the future is the one that is in effect. This is similar to but more general than the scheduled discount. It is more general, because it provides a means to vary the fee agreement for each time period.

Grammar element 1525 **"Markup-Spec: = Markup: percentage To: Account-ID"** is provided for adding a percentage to the fees already being charged. For example, a 5% markup means that a fee of 5% of cumulative fee so far will be allocated to the distributor. A markup specification can be applied to all of the other kinds of fee specifications. It is typically used in a shell provided by a distributor. It refers to fees associated with d-blocks that are parts of the current d-block. This might be a convenient specification for use in taxes, or in distributor overhead.

REPOSITORY TRANSACTIONS

When a user requests access to a digital work, the repository will initiate various transactions. The combination of transactions invoked will depend on the specifications assigned for a usage right. There are three basic types of transactions, Session Initiation Transactions, Financial Transactions and Usage Transactions. Generally, session initiation transactions are initiated first to establish a valid session. When a valid session is established, transactions corresponding to the various usage rights are invoked. Finally, request specific transactions are performed.

Transactions occur between two repositories (one acting as a server), between a repository and a document playback platform (e.g. for executing or viewing), between a repository and a credit server or between a repository and an authorization server. When transactions occur between more than one repository, it is assumed that there is a reliable communication channel between the repositories. For example, this could be a TCP/IP channel or any other commercially available channel that has built-in capabilities for detecting and correcting transmission errors. However, it is not assumed that the communication channel is secure. Provisions for security and privacy are part of the requirements for specifying and implementing repositories and thus form the need for various transactions.

Message Transmission

Transactions require that there be some communication between repositories. Communication between repositories occurs in units termed as messages. Because the communication line is assumed to be unsecure, all communications with repositories that are above the lowest security class are encrypted utilizing a public key encryption technique. Public key encryption is a well known technique in the encryption arts. The term key refers to a numeric code that is used with encryption and decryption algorithms. Keys come in pairs, where "writing keys" are used to encrypt data and "checking keys" are used to decrypt data. Both writing and checking keys may be public or private. Public keys are those that are distributed to others. Private keys are maintained in confidence.

Key management and security is instrumental in the success of a public key encryption system. In the currently preferred embodiment, one or more master repositories maintain the keys and create the identification certificates used by the repositories.

When a sending repository transmits a message to a receiving repository, the sending repository encrypts all of its data using the public writing key of the receiving repository. The sending repository includes its name, the name of the receiving repository, a session identifier such as a nonce (described below), and a message counter in each message.

In this way, the communication can only be read (to a high probability) by the receiving repository, which holds the private checking key for decryption. The auxiliary data is used to guard against various replay attacks to security. If messages ever arrive with the wrong counter or an old nonce, the repositories can assume that someone is interfering with communication and the transaction terminated.

The respective public keys for the repositories to be used for encryption are obtained in the registration transaction described below.

DOCID <EP_____0715247A1 I. >.

Session Initiation Transactions

A usage transaction is carried out in a session between repositories. For usage transactions involving more than one repository, or for financial transactions between a repository and a credit server, a registration transaction is performed. A second transaction termed a login transaction, may also be needed to initiate the session. The goal of the registration transaction is to establish a secure channel between two repositories who know each others identities. As it is assumed that the communication channel between the repositories is reliable but not secure, there is a risk that a non-repository may mimic the protocol in order to gain illegitimate access to a repository.

The registration transaction between two repositories is described with respect to Figures 16 and 17. The steps described are from the perspective of a "repository-1" registering its identity with a "repository-2". The registration must be symmetrical so the same set of steps will be repeated for repository-2 registering its identity with repository-1. Referring to Figure 16, repository-1 first generates an encrypted registration identifier, step 1601 and then generates a registration message, step 1602. A registration message is comprised of an identifier of a master repository, the identification certificate for the repository-1 and an encrypted random registration identifier. The identification certificate is encrypted by the master repository in its private key and attests to the fact that the repository (here repository-1) is a bona fide repository. The identification certificate also contains a public key for the repository, the repository security level and a timestamp (indicating a time after which the certificate is no longer valid.) The registration identifier is a number generated by the repository for this registration. The registration identifier is unique to the session and is encrypted in repository-1's private key. The registration identifier is used to improve security of authentication by detecting certain kinds of communications based attacks. Repository-1 then transmits the registration message to repository-2, step 1603.

Upon receiving the registration message, repository-2 determines if it has the needed public key for the master repository, step 1604. If repository-2 does not have the needed public key to decrypt the identification certificate, the registration transaction terminates in an error, step 1618.

Assuming that repository-2 has the proper public key the identification certificate is decrypted, step 1605. Repository-2 saves the encrypted registration identifier, step 1606, and extracts the repository identifier, step 1607. The extracted repository identifier is checked against a "hotlist" of compromised document repositories, step 1608. In the currently preferred embodiment, each repository will contain "hotlists" of compromised repositories. If the repository is on the "hotlist", the registration transaction terminates in an error per step 1618. Repositories can be removed from the hotlist when their certificates expire, so that the list does not need to grow without bound. Also, by keeping a short list of hotlist certificates that it has previously received, a repository can avoid the work of actually going through the list. These lists would be encrypted by a master repository. A minor variation on the approach to improve efficiency would have the repositories first exchange lists of names of hotlist certificates, ultimately exchanging only those lists that they had not previously received. The "hotlists" are maintained and distributed by Master repositories.

Note that rather than terminating in error, the transaction could request that another registration message be sent based on an identification certificate created by another master repository. This may be repeated until a satisfactory identification certificate is found, or it is determined that trust cannot be established.

Assuming that the repository is not on the hotlist, the repository identification needs to be verified. In other words, repository-2 needs to validate that the repository on the other end is really repository-1. This is termed performance testing and is performed in order to avoid invalid access to the repository via a counterfeit repository replaying a recording of a prior session initiation between repository-1 and repository-2. Performance testing is initiated by repository-2 generating a performance message, step 1609. The performance message consists of a nonce, the names of the respective repositories, the time and the registration identifier received from repository-1. A nonce is a generated message based on some random and variable information (e.g. the time or the temperature.) The nonce is used to check whether repository-1 can actually exhibit correct encrypting of a message using the private keys it claims to have, on a message that it has never seen before. The performance message is encrypted using the public key specified in the registration message of repository-1. The performance message is transmitted to repository-1, step 1610, where it is decrypted by repository-1 using its private key, step 1611. Repository-1 then checks to make sure that the names of the two repositories are correct, step 1612, that the time is accurate, step 1613 and that the registration identifier corresponds to the one it sent, step 1614. If any of these tests fails, the transaction is terminated per step 1616. Assuming that the tests are passed, repository-1 transmits the nonce to repository-2 in the clear, step 1615. Repository-2 then compares the received nonce to the original nonce, step 1617. If they are not identical, the registration transaction terminates in an error per step 1618. If they are the same, the registration transaction has successfully completed.

At this point, assuming that the transaction has not terminated, the repositories exchange messages containing session keys to be used in all communications during the session and synchronize their clocks. Figure 17 illustrates the session information exchange and clock synchronization steps (again from the perspective of repository-1.) Referring to Figure 17, repository-1 creates a session key pair, step 1701. A first key is kept private and is used by repository-1 to encrypt messages. The second key is a public key used by repository-2 to decrypt messages. The

DOCID: <EP.....0715247A1 I >

second key is encrypted using the public key of repository-2, step 1702 and is sent to repository-2, step 1703. Upon receipt, repository-2 decrypts the second key, step 1704. The second key is used to decrypt messages in subsequent communications. When each repository has completed this step, they are both convinced that the other repository is bona fide and that they are communicating with the original. Each repository has given the other a key to be used in decrypting further communications during the session. Since that key is itself transmitted in the public key of the receiving repository only it will be able to decrypt the key which is used to decrypt subsequent messages.

After the session information is exchanged, the repositories must synchronize their clocks. Clock synchronization is used by the repositories to establish an agreed upon time base for the financial records of their mutual transactions. Referring back to Figure 17, repository-2 initiates clock synchronization by generating a time stamp exchange message, step 1705, and transmits it to repository-1, step 1706. Upon receipt, repository-1 generates its own time stamp message, step 1707 and transmits it back to repository-2, step 1708. Repository-2 notes the current time, step 1709 and stores the time received from repository-1, step 1710. The current time is compared to the time received from repository-1, step 1711. The difference is then checked to see if it exceeds a predetermined tolerance (e.g. one minute), step 1712. If it does, repository-2 terminates the transaction as this may indicate tampering with the repository, step 1713. If not repository-2 computes an adjusted time delta, step 1714. The adjusted time delta is the difference between the clock time of repository-2 and the average of the times from repository-1 and repository-2.

To achieve greater accuracy, repository-2 can request the time again up to a fixed number of times (e.g. five times), repeat the clock synchronization steps, and average the results.

A second session initiation transaction is a Login transaction. The Login transaction is used to check the authenticity of a user requesting a transaction. A Login transaction is particularly prudent for the authorization of financial transactions that will be charged to a credit server. The Login transaction involves an interaction between the user at a user interface and the credit server associated with a repository. The information exchanged here is a login string supplied by the repository/credit server to identify itself to the user, and a Personal Identification Number (PIN) provided by the user to identify himself to the credit server. In the event that the user is accessing a credit server on a repository different from the one on which the user interface resides, exchange of the information would be encrypted using the public and private keys of the respective repositories.

Billing Transactions

Billing Transactions are concerned with monetary transactions with a credit server. Billing Transactions are carried out when all other conditions are satisfied and a usage fee is required for granting the request. For the most part, billing transactions are well understood in the state of the art. These transactions are between a repository and a credit server, or between a credit server and a billing clearinghouse. Briefly, the required transactions include the following:

- Registration and LOG IN transactions by which the repository and user establish their bona fides to a credit server. These transactions would be entirely internal in cases where the repository and credit server are implemented as a single system.
- Registration and LOGIN transactions, by which a credit server establishes its bona fides to a billing clearinghouse.
- An Assign-fee transaction to assign a charge. The information in this transaction would include a transaction identifier, the identities of the repositories in the transaction, and a list of charges from the parts of the digital work. If there has been any unusual event in the transaction such as an interruption of communications, that information is included as well.
- A Begin-charges transaction to assign a charge. This transaction is much the same as an assign-fee transaction except that it is used for metered use. It includes the same information as the assign-fee transaction as well as the usage fee information. The credit-server is then responsible for running a clock.
- An End-charges transaction to end a charge for metered use. (In a variation on this approach, the repositories would exchange periodic charge information for each block of time.)
- A report-charges transaction between a personal credit server and a billing clearinghouse. This transaction is invoked at least once per billing period. It is used to pass along information about charges. On debit and credit cards, this transaction would also be used to update balance information and credit limits as needed.

All billing transactions are given a transaction ID and are reported to the credit severs by both the server and the client. This reduces possible loss of billing information if one of the parties to a transaction loses a banking card and provides a check against tampering with the system.

Usage Transactions

After the session initiation transactions have been completed, the usage request may then be processed. To sim-

plify the description of the steps carried out in processing a usage request, the term requester is used to refer to a repository in the requester mode which is initiating a request, and the term server is used to refer to a repository in the server mode and which contains the desired digital work. In many cases such as requests to print or view a work, the requester and server may be the same device and the transactions described in the following would be entirely internal. In such instances, certain transaction steps, such as the registration transaction, need not be performed.

There are some common steps that are part of the semantics of all of the usage rights transactions. These steps are referred to as the common transaction steps. There are two sets -- the "opening" steps and the "closing" steps. For simplicity, these are listed here rather than repeating them in the descriptions of all of the usage rights transactions.

Transactions can refer to a part of a digital work, a complete digital work, or a Digital work containing other digital works. Although not described in detail herein, a transaction may even refer to a folder comprised of a plurality of digital works. The term "work" is used to refer to what ever portion or set of digital works is being accessed.

Many of the steps here involve determining if certain conditions are satisfied. Recall that each usage right may have one or more conditions which must be satisfied before the right can be exercised. Digital works have parts and parts have parts. Different parts can have different rights and fees. Thus, it is necessary to verify that the requirements are met for ALL of the parts that are involved in a transaction For brevity, when reference is made to checking whether the rights exist and conditions for exercising are satisfied, it is meant that all such checking takes place for each of the relevant parts of the work

Figure 18 illustrates the initial common opening and closing steps for a transaction. At this point it is assumed that registration has occurred and that a "trusted" session is in place. General tests are tests on usage rights associated with the folder containing the work or some containing folder higher in the file system hierarchy. These tests correspond to requirements imposed on the work as a consequence of its being on the particular repository, as opposed to being attached to the work itself. Referring to Figure 18, prior to initiating a usage transaction, the requester performs any general tests that are required before the right associated with the transaction can be exercised, step, 1801. For example, install, uninstall and delete rights may be implemented to require that a requester have an authorization certificate before the right can be exercised. Another example is the requirement that a digital ticket be present and punched before a digital work may be copied to a requester. If any of the general tests fail, the transaction is not initiated, step, 1802. Assuming that such required tests are passed, upon receiving the usage request, the server generates a transaction identifier that is used in records or reports of the transaction, step 1803. The server then checks whether the digital work has been granted the right corresponding to the requested transaction, step 1804. If the digital work has not been granted the right corresponding to the request, the transaction terminates, step 1805. If the digital work has been granted the requested right, the server then determines if the various conditions for exercising the right are satisfied. Time based conditions are examined, step 1806. These conditions are checked by examining the time specification for the the version of the right. If any of the conditions are not satisfied, the transaction terminates per step 1805.

Assuming that the time based conditions are satisfied, the server checks security and access conditions, step 1807. Such security and access conditions are satisfied if: 1) the requester is at the specified security class, or a higher security class, 2) the server satisfies any specified authorization test and 3) the requester satisfies any specified authorization tests and has any required digital tickets. If any of the conditions are not satisfied, the transaction terminates per step 1805.

Assuming that the security and access conditions are all satisfied, the server checks the copy count condition, step 1808. If the copy count equals zero, then the transaction cannot be completed and the transaction terminates per step 1805.

Assuming that the copy count does not equal zero, the server checks if the copies in use for the requested right is greater than or equal to any copy count for the requested right (or relevant parts), step 1809. If the copies in use is greater than or equal to the copy count, this indicates that usage rights for the version of the transaction have been exhausted. Accordingly, the server terminates the transaction, step 1805. If the copy count is less than the copies in use for the transaction the transaction can continue, and the copies in use would be incremented by the number of digital works requested in the transaction, step 1810

The server then checks if the digital work has a "Loan" access right, step 1811. The "Loan" access right is a special case since remaining rights may be present even though all copies are loaned out. If the digital work has the "Loan" access right, a check is made to see if all copies have been loaned out, step 1812. The number of copies that could be loaned is the sum of the Copy-Counts for all of the versions of the loan right of the digital work. For a composite work, the relevant figure is the minimal such sum of each of the components of the composite work. If all copies have been loaned out, the remaining rights are determined, step 1813. The remaining-rights is determined from the remaining rights specifications from the versions of the Loan right. If there is only one version of the Loan right, then the determination is simple. The remaining rights are the ones specified in that version of the Loan right, or none if Remaining-Rights is not specified. If there are multiple versions of the Loan right and all copies of all of the versions are loaned out, then the remaining rights is taken as the minimum set (intersection) of remaining rights across all of the versions of the loan right. The server then determines if the requested right is in the set of remaining rights, step 1814. If the

SDOCID: <EP 0715247A1 I >

requested right is not in the set of remaining rights, the server terminates the transaction, step 1805.

If Loan is not a usage right for the digital work or if all copies have not been loaned out or the requested right is in the set of remaining rights, fee conditions for the right are then checked, step 1815. This will initiate various financial transactions between the repository and associated credit server. Further, any metering of usage of a digital work will commence. If any financial transaction fails, the transaction terminates per step 1805.

It should be noted that the order in which the conditions are checked need not follow the order of steps 1806-1815.

At this point, right specific steps are now performed and are represented here as step 1816. The right specific steps are described in greater detail below.

The common closing transaction steps are now performed. Each of the closing transaction steps are performed by the server after a successful completion of a transaction. Referring back to Figure 18, the copies in use value for the requested right is decremented by the number of copies involved in the transaction, step 1817. Next, if the right had a metered usage fee specification, the server subtracts the elapsed time from the Remaining-Use-Time associated with the right for every part involved in the transaction, step 1818. Finally, if there are fee specifications associated with the right, the server initiates End-Charge financial transaction to confirm billing, step 1819.

Transmission Protocol

An important area to consider is the transmission of the digital work from the server to the requester. The transmission protocol described herein refers to events occurring after a valid session has been created. The transmission protocol must handle the case of disruption in the communications between the repositories. It is assumed that interference such as injecting noise on the communication channel can be detected by the integrity checks (e.g., parity, checksum, etc.) that are built into the transport protocol and are not discussed in detail herein.

The underlying goal in the transmission protocol is to preclude certain failure modes, such as malicious or accidental interference on the communications channel. Suppose, for example, that a user pulls a card with the credit server at a specific time near the end of a transaction. There should not be a vulnerable time at which "pulling the card" causes the repositories to fail to correctly account for the number of copies of the work that have been created. Restated, there should be no time at which a party can break a connection as a means to avoid payment after using a digital work.

If a transaction is interrupted (and fails), both repositories restore the digital works and accounts to their state prior to the failure, modulo records of the failure itself.

Figure 19 is a state diagram showing steps in the process of transmitting information during a transaction. Each box represents a state of a repository in either the server mode (above the central dotted line 1901) or in the requester mode (below the dotted line 1901). Solid arrows stand for transitions between states. Dashed arrows stand for message communications between the repositories. A dashed message arrow pointing to a solid transition arrow is interpreted as meaning that the transition takes place when the message is received. Unlabeled transition arrows take place unconditionally. Other labels on state transition arrows describe conditions that trigger the transition.

Referring now to Figure 19, the server is initially in a state 1902 where a new transaction is initiated via start message 1903. This message includes transaction information including a transaction identifier and a count of the blocks of data to be transferred. The requester, initially in a wait state 1904 then enters a data wait state 1905.

The server enters a data transmit state 1906 and transmits a block of data 1907 and then enters a wait for acknowledgement state 1908. As the data is received, the requester enters a data receive state 1909 and when the data blocks are completely received it enters an acknowledgement state 1910 and transmits an Acknowledgement message 1911 to the server.

If there are more blocks to send, the server waits until receiving an Acknowledgement message from the requester. When an Acknowledgment message is received it sends the next block to the requester and again waits for acknowledgement. The requester also repeats the same cycle of states.

If the server detects a communications failure before sending the last block, it enters a cancellation state 1912 wherein the transaction is cancelled. Similarly, if the requester detects a communications failure before receiving the last block it enters a cancellation state 1913.

If there are no more blocks to send, the server commits to the transaction and waits for the final Acknowledgement in state 1914. If there is a communications failure before the server receives the final Acknowledgement message, it still commits to the transaction but includes a report about the event to its credit server in state 1915. This report serves two purposes. It will help legitimize any claims by a user of having been billed for receiving digital works that were not completely received. Also it helps to identify repositories and communications lines that have suspicious patterns of use and interruption. The server then enters its completion state 1916.

On the requester side, when there are no more blocks to receive, the requester commits to the transaction in state 1917. If the requester detects a communications failure at this state, it reports the failure to its credit server in state 1918, but still commits to the transaction. When it has committed, it sends an acknowledgement message to the server. The server then enters its completion state 1919.

ISDOCID <EP 0715247A1 I >

The key property is that both the server and the requester cancel a transaction if it is interrupted before all of the data blocks are delivered, and commits to it if all of the data blocks have been delivered.

There is a possibility that the server will have sent all of the data blocks (and committed) but the requester will not have received all of them and will cancel the transaction. In this case, both repositories will presumably detect a communications failure and report it to their credit server. This case will probably be rare since it depends on very precise timing of the communications failure. The only consequence will be that the user at the requester repository may want to request a refund from the credit services -- and the case for that refund will be documented by reports by both repositories.

To prevent loss of data, the server should not delete any transferred digital work until receiving the final acknowledgement from the requester. But it also should not use the file. A well known way to deal with this situation is called "two-phase commit" or 2PC.

Two-phase commit works as follows. The first phase works the same as the method described above. The server sends all of the data to the requester. Both repositories mark the transaction (and appropriate files) as uncommitted. The server sends a ready-to-commit message to the requester. The requester sends back an acknowledgement. The server then commits and sends the requester a commit message. When the requester receives the commit message, it commits the file.

If there is a communication failure or other crash, the requester must check back with the server to determine the status of the transaction. The server has the last word on this. The requester may have received all of the data, but if it did not get the final message, it has not committed. The server can go ahead and delete files (except for transaction records) once it commits, since the files are known to have been fully transmitted before starting the 2PC cycle.

There are variations known in the art which can be used to achieve the same effect. For example, the server could use an additional level of encryption when transmitting a work to a client. Only after the client sends a message acknowledging receipt does it send the key. The client then agrees to pay for the digital work. The point of this variation is that it provides a clear audit trail that the client received the work. For trusted systems, however, this variation adds a level of encryption for no real gain in accountability.

The transaction for specific usage rights are now discussed.

The Copy Transaction

A Copy transaction is a request to make one or more independent copies of the work with the same or lesser usage rights. Copy differs from the extraction right discussed later in that it refers to entire digital works or entire folders containing digital works. A copy operation cannot be used to remove a portion of a digital work.

- The requester sends the server a message to initiate the Copy Transaction. This message indicates the work to be copied, the version of the copy right to be used for the transaction, the destination address information (location in a folder) for placing the work, the file data for the work (including its size), and the number of copies requested.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the client according to the transmission protocol. If a Next-Set-Of-Rights has been provided in the version of the right, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In any event, the Copy-Count field for the copy of the digital work being sent right is set to the number-of-copies requested.
- The requester records the work contents, data, and usage rights and stores the work. It records the date and time that the copy was made in the properties of the digital work.
- The repositories perform the common closing transaction steps

The Transfer Transaction

A Transfer transaction is a request to move copies of the work with the same or lesser usage rights to another repository. In contrast with a copy transaction, this results in removing the work copies from the server.

- The requester sends the server a message to initiate the Transfer Transaction. This message indicates the work to be transferred, the version of the transfer right to be used in the transaction, the destination address information for placing the work, the file data for the work, and the number of copies involved.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In either case, the Copy-Count field for the transmitted rights are set to the

number-of-copies requested
- The requester records the work contents, data, and usage rights and stores the work.
- The server decrements its copy count by the number of copies involved in the transaction.
- The repositories perform the common closing transaction steps.
- If the number of copies remaining in the server is now zero, it erases the digital work from its memory.

The Loan Transaction

A loan transaction is a mechanism for loaning copies of a digital work. The maximum duration of the loan is determined by an internal parameter of the digital work. Works are automatically returned after a predetermined time period.

- The requester sends the server a message to initiate the Transfer Transaction. This message indicates the work to be loaned, the version of the loan right to be used in the transaction, the destination address information for placing the work, the number of copies involved, the file data for the work, and the period of the loan.
- The server checks the validity of the requested loan period, and ends with an error if the period is not valid. Loans for a loaned copy cannot extend beyond the period of the original loan to the server.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted, as modified to reflect the loan period.
- The requester records the digital work contents, data, usage rights, and loan period and stores the work.
- The server updates the usage rights information in the digital work to reflect the number of copies loaned out.
- The repositories perform the common closing transaction steps.
- The server updates the usage rights data for the digital work. This may preclude use of the work until it is returned from the loan. The user on the requester platform can now use the transferred copies of the digital work. A user accessing the original repository cannot use the digital work, unless there are copies remaining. What happens next depends on the order of events in time.

Case 1. If the time of the loan period is not yet exhausted and the requester sends the repository a Return message.

- The return message includes the requester identification, and the transaction ID.
- The server decrements the copies-in-use field by the number of copies that were returned. (If the number of digital works returned is greater than the number actually borrowed, this is treated as an error.) This step may now make the work available at the server for other users.
- The requester deactivates its copies and removes the contents from its memory.

Case 2. If the time of the loan period is exhausted and the requester has not yet sent a Return messages

- The server decrements the copies-in-use field by the number digital works that were borrowed.
- The requester automatically deactivates its copies of the digital work. It terminates all current uses and erases the digital work copies from memory. One question is why a requester would ever return a work earlier than the period of the loan, since it would be returned automatically anyway. One reason for early return is that there may be a metered fee which determines the cost of the loan. Returning early may reduce that fee

The Play Transaction

A play transaction is a request to use the contents of a work. Typically, to "play" a work is to send the digital work through some kind of transducer, such as a speaker or a display device. The request implies the intention that the contents will not be communicated digitally to any other system. For example, they will not be sent to a printer, recorded on any digital medium, retained after the transaction or sent to another repository.

This term "play" is natural for examples like playing music, playing a movie, or playing a video game. The general form of play means that a "player" is used to use the digital work. However, the term play covers all media and kinds of recordings. Thus one would "play" a digital work, meaning, to render it for reading, or play a computer program, meaning to execute it. For a digital ticket the player would be a digital ticket agent.

3DOCID: <EP _____ 0715247A1_!_>

- The requester sends the server a message to initiate the play transaction. This message indicates the work to be played, the version of the play right to be used in the transaction, the identity of the player being used, and the file data for the work.
- The server checks the validity of the player identification and the compatibility of the player identification with the player specification in the right. It ends with an error if these are not satisfactory.
- The repositories perform the common opening transaction steps.
- The server and requester read and write the blocks of data as requested by the player according to the transmission protocol. The requester plays the work contents, using the player.
- When the player is finished, the player and the requester remove the contents from their memory
- The repositories perform the common closing transaction steps.

The Print Transaction

A Print transaction is a request to obtain the contents of a work for the purpose of rendering them on a "printer." We use the term "printer" to include the common case of writing with ink on paper. However, the key aspect of "printing" in our use of the term is that it makes a copy of the digital work in a place outside of the protection of usage rights. As with all rights, this may require particular authorization certificates.

Once a digital work is printed, the publisher and user are bound by whatever copyright laws are in effect. However, printing moves the contents outside the control of repositories. For example, absent any other enforcement mechanisms, once a digital work is printed on paper, it can be copied on ordinary photocopying machines without intervention by a repository to collect usage fees. If the printer to a digital disk is permitted, then that digital copy is outside of the control of usage rights. Both the creator and the user know this, although the creator does not necessarily give tacit consent to such copying, which may violate copyright laws

- The requester sends the server a message to initiate a Print transaction. This message indicates the work to be played, the identity of the printer being used, the file data for the work, and the number of copies in the request.
- The server checks the validity of the printer identification and the compatibility of the printer identification with the printer specification in the right. It ends with an error if these are not satisfactory.
- The repositories perform the common opening transaction steps.
- The server transmits blocks of data according to the transmission protocol.
- The requester prints the work contents, using the printer.
- When the printer is finished, the printer and the requester remove the contents from their memory.
- The repositories perform the common closing transaction steps.

The Backup Transaction

A Backup transaction is a request to make a backup copy of a digital work, as a protection against media failure. In the context of repositories, secure backup copies differ from other copies in three ways: (1) they are made under the control of a Backup transaction rather than a Copy transaction, (2) they do not count as regular copies, and (3) they are not usable as regular copies. Generally, backup copies are encrypted.

Although backup copies may be transferred or copied, depending on their assigned rights, the only way to make them useful for playing, printing or embedding is to restore them.

The output of a Backup operation is both an encrypted data file that contains the contents and description of a work, and a restoration file with an encryption key for restoring the encrypted contents. In many cases, the encrypted data file would have rights for "printing" it to a disk outside of the protection system, relying just on its encryption for security. Such files could be stored anywhere that was physically safe and convenient. The restoration file would be held in the repository. This file is necessary for the restoration of a backup copy. It may have rights for transfer between repositories.

- The requester sends the server a message to initiate a backup transaction. This message indicates the work to be backed up, the version of the backup right to be used in the transaction, the destination address information for placing the backup copy, the file data for the work.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, a set of default rights for backup files of the original are transmitted by the server.
- The requester records the work contents, data, and usage rights. It then creates a one-time key and encrypts the contents file. It saves the key information in a restoration file.

SDOCID: <EP 0715247A1 I >

- The repositories perform the common closing transaction steps.

In some cases, it is convenient to be able to archive the large, encrypted contents file to secure offline storage, such as a magneto-optical storage system or magnetic tape. This creation of a non-repository archive file is as secure as the encryption process. Such non-repository archive storage is considered a form of "printing" and is controlled by a print right with a specified "archive-printer." An archive-printer device is programmed to save the encrypted contents file (but not the description file) offline in such a way that it can be retrieved.

The Restore Transaction

A Restore transaction is a request to convert an encrypted backup copy of a digital work into a usable copy. A restore operation is intended to be used to compensate for catastrophic media failure. Like all usage rights, restoration rights can include fees and access tests including authorization checks.

- The requester sends the server a message to initiate a Restore transaction. This message indicates the work to be restored, the version of the restore right for the transaction, the destination address information for placing the work, and the file data for the work.
- The server verifies that the contents file is available (i.e. a digital work corresponding to the request has been backed-up.) If it is not, it ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server retrieves the key from the restoration file. It decrypts the work contents, data, and usage rights.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, a set of default rights for backup files of the original are transmitted by the server.
- The requester stores the digital work.
- The repositories perform the common closing transaction steps.

The Delete Transaction

A Delete transaction deletes a digital work or a number of copies of a digital work from a repository. Practically all digital works would have delete rights.

- The requester sends the server a message to initiate a delete transaction. This message indicates the work to be deleted , the version of the delete right for the transaction.
- The repositories perform the common opening transaction steps.
- The server deletes the file, erasing it from the file system.
- The repositories perform the common closing transaction steps

The Directory Transaction

A Directory transaction is a request for information about folders, digital works, and their parts. This amounts to roughly the same idea as protection codes in a conventional file system like TENEX, except that it is generalized to the full power of the access specifications of the usage rights language.

The Directory transaction has the important role of passing along descriptions of the rights and fees associated with a digital work. When a user wants to exercise a right, the user interface of his repository implicitly makes a directory request to determine the versions of the right that are available. Typically these are presented to the user -- such as with different choices of billing for exercising a right. Thus, many directory transactions are invisible to the user and are exercised as part of the normal process of exercising all rights.

- The requester sends the server a message to initiate a Directory transaction. This message indicates the file or folder that is the root of the directory request and the version of the directory right used for the transaction.
- The server verifies that the information is accessible to the requester. In particular, it does not return the names of any files that have a HIDE-NAME status in their directory specifications, and it does not return the parts of any folders or files that have HIDE-PARTS in their specification. If the information is not accessible, the server ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server sends the requested data to the requester according to the transmission protocol.
- The requester records the data.

- The repositories perform the common closing transaction steps.

The Folder Transaction

A Folder transaction is a request to create or rename a folder, or to move a work between folders. Together with Directory rights, Folder rights control the degree to which organization of a repository can be accessed or modified from another repository.

- The requester sends the server a message to initiate a Folder transaction. This message indicates the folder that is the root of the folder request, the version of the folder right for the transaction, an operation, and data. The operation can be one of create, rename, and move file. The data are the specifications required for the operation, such as a specification of a folder or digital work and a name.
- The repositories perform the common opening transaction steps.
- The server performs the requested operation -- creating a folder, renaming a folder, or moving a work between folders.
- The repositories perform the common closing transaction steps

The Extract Transaction

A extract transaction is a request to copy a part of a digital work and to create a new work containing it. The extraction operation differs from copying in that it can be used to separate a part of a digital work from d-blocks or shells that place additional restrictions or fees on it. The extraction operation differs from the edit operation in that it does not change the contents of a work, only its embedding in d-blocks. Extraction creates a new digital work.

- The requester sends the server a message to initiate an Extract transaction. This message indicates the part of the work to be extracted, the version of the extract right to be used in the transaction, the destination address information for placing the part as a new work, the file data for the work, and the number of copies involved.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested.
- The requester records the contents, data, and usage rights and stores the work. It records the date and time that new work was made in the properties of the work.
- The repositories perform the common closing transaction steps.

The Embed Transaction

An embed transaction is a request to make a digital work become a part of another digital work or to add a shell d-block to enable the adding of fees by a distributor of the work.

- The requester sends the server a message to initiate an Embed transaction. This message indicates the work to be embedded, the version of the embed right to be used in the transaction, the destination address information for placing the part as a a work, the file data for the work, and the number of copies involved.
- The server checks the control specifications for all of the rights in the part and the destination. If they are incompatible, the server ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested.
- The requester records the contents, data, and usage rights and embeds the work in the destination file.
- The repositories perform the common closing transaction steps.

The Edit Transaction

An Edit transaction is a request to make a new digital work by copying, selecting and modifying portions of an existing digital work. This operation can actually change the contents of a digital work. The kinds of changes that are permitted depend on the process being used. Like the extraction operation, edit operates on portions of a digital work. In contrast with the extract operation, edit does not affect the rights or location of the work. It only changes the contents.

(19)EPO/OEP 0715247A1 1 2

The kinds of changes permitted are determined by the type specification of the processor specified in the rights. In the currently preferred embodiment, an edit transaction changes the work itself and does not make a new work. However, it would be a reasonable variation to cause a new copy of the work to be made.

- The requester sends the server a message to initiate an Edit transaction. This message indicates the work to be edited, the version of the edit right to be used in the transaction, the file data for the work (including its size), the process-ID for the process, and the number of copies involved.
- The server checks the compatibility of the process-ID to be used by the requester against any process-ID specification in the right. If they are incompatible, it ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The requester uses the process to change the contents of the digital work as desired. (For example, it can select and duplicate parts of it; combine it with other information; or compute functions based on the information. This can amount to editing text, music, or pictures or taking whatever other steps are useful in creating a derivative work.)
- The repositories perform the common closing transaction steps.

The edit transaction is used to cover a wide range of kinds of works. The category describes a process that takes as its input any portion of a digital work and then modifies the input in some way. For example, for text, a process for editing the text would require edit rights. A process for "summarizing" or counting words in the text would also be considered editing. For a music file, processing could involve changing the pitch or tempo, or adding reverberations, or any other audio effect. For digital video works, anything which alters the image would require edit rights. Examples would be colorizing, scaling, extracting still photos, selecting and combining frames into story boards, sharpening with signal processing, and so on.

Some creators may want to protect the authenticity of their works by limiting the kinds of processes that can be performed on them. If there are no edit rights, then no processing is allowed at all. A processor identifier can be included to specify what kind of process is allowed. If no process identifier is specified, then arbitrary processors can be used. For an example of a specific process, a photographer may want to allow use of his photograph but may not want it to be colorized. A musician may want to allow extraction of portions of his work but not changing of the tonality.

Authorization Transactions

There are many ways that authorization transactions can be defined. In the following, our preferred way is to simply define them in terms of other transactions that we already need for repositories. Thus, it is convenient sometimes to speak of "authorization transactions," but they are actually made up of other transactions that repositories already have.

A usage right can specify an authorization-ID, which identifies an authorization object (a digital work in a file of a standard format) that the repository must have and which it must process. The authorization is given to the generic authorization (or ticket) server of the repository which begins to interpret the authorization.

As described earlier, the authorization contains a server identifier, which may just be the generic authorization server or it may be another server. When a remote authorization server is required, it must contain a digital address. It may also contain a digital certificate.

If a remote authorization server is required, then the authorization process first performs the following steps:

- The generic authorization server attempts to set up the communications channel. (If the channel cannot be set up, then authorization fails with an error.)
- When the channel is set up, it performs a registration process with the remote repository. (If registration fails, then the authorization fails with an error.)
- When registration is complete, the generic authorization server invokes a "Play" transaction with the remote repository, supplying the authorization document as the digital work to be played, and the remote authorization server (a program) as the "player." (If the player cannot be found or has some other error, then the authorization fails with an error.)
- The authorization server then "plays" the authorization. This involves decrypting it using either the public key of the master repository that issued the certificate or the session key from the repository that transmitted it. The authorization server then performs various tests. These tests vary according to the authorization server. They include such steps as checking issue and validity dates of the authorization and checking any hot-lists of known invalid authorizations. The authorization server may require carrying out any other transactions on the repository as well, such as checking directories, getting some person to supply a password, or playing some other digital work. It may also invoke some special process for checking information about locations or recent events. The "script" for such steps is contained within the authorization server.
- If all of the required steps are completed satisfactorily, the authorization server completes the transaction normally.

SOCCID <EP.....0715247A1_I_>

signaling that authorization is granted.

The Install Transaction

An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed.

- The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size).
- The repositories perform the common opening transaction steps.
- The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error.
- The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.)
- The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.)
- The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository the installation transaction ends with an error. (This step checks platform compatibility.)
- The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions.
- The repositories perform the common closing transaction steps.

The Uninstall Transaction

An Uninstall transaction is a request to remove software from a repository. Since uncontrolled or incorrect removal of software from a repository could compromise its behavioral integrity, this step is controlled.

- The requester sends the server an Uninstall message. This message indicates the work to be uninstalled, the version of the Uninstall right being invoked, and the file data for the work (including its size).
- The repositories perform the common opening transaction steps
- The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error.
- The requester checks whether the software is installed. If the software is not installed, the transaction ends with an error.
- The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step authenticates the certification of the software, including the script for uninstalling it.)
- The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.)
- The requester retrieves the instructions in the uninstallation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error.
- The repositories perform the common closing transaction steps.

Claims

1. A system for controlling the distribution and use of digital works comprising:

   means for attaching one or more usage rights to a digital work, each of said one or more usage rights specifying

EDOCID <EP 0715247A1 I >

a particular instance of how said digital work may be used or distributed, each of said usage rights being capable of specifying a digital ticket, the possession of said digital ticket being a condition on the exercise of a right specifying said digital ticket;
a plurality of repositories for storing and exchanging digital works, each of said plurality of repositories comprising;
storage means for storing digital works, their attached usage rights, and digital tickets;
transaction processing means having a requester mode of operation for requesting access to a digital work, said request specifying a usage right, and a server mode of operation for processing requests to access said requested digital work based on said usage right specified in said request, the usage rights attached to said digital work, and digital tickets associated with said usage rights;
a generic ticket agent for punching digital tickets to indicate that an associated usage right has been exercised; and
a coupling means for coupling to another of said plurality of repositories across a communications medium.

2. The system as recited in Claim 1 wherein a usage right may specify a special ticket agent for punching a specified digital ticket and said system is further comprised of one or more authorization repositories containing special ticket agents for punching digital tickets.

3. The system as recited in Claim 1 wherein said digital ticket is comprised of a first timestamp part for indicating a time when said digital ticket was copied and a second timestamp part for indicating a time when said digital ticket was punched.

4. The system as recited in Claim 3 wherein said generic ticket agent is further comprised of a means for determining if a digital ticket is valid and timestamping means for timestamping digital tickets when they are punched.

5. A method for controlling access to digital works in a computer controlled system for the distribution of digital works, comprising the steps of:

a) creating a digital work and a digital ticket for said digital work;
b) attaching a usage right to said digital work, said usage right specifying said digital ticket;
c) storing said digital work in a first repository;
d) storing said digital ticket in a second repository;
e) a third repository acquiring said digital ticket from said second repository;
f) said third repository transmitting a request to access said digital work to said first repository, said request specifying said usage right;
g) said first repository determining if said third repository has said digital ticket;
h) said third repository presenting said digital ticket to said first repository,
i) said first repository transmitting said digital work to said third repository; and
j) said first repository punching said digital ticket.

6. The method as recited in Claim 5 wherein said step of a third repository acquiring said digital ticket from said second repository is further comprised of the step of said third repository copying said digital ticket from said second repository for a fee.

7. The method as recited in Claim 6 wherein said digital ticket is comprised of a first timestamp part for storing a time when said digital ticket was copied and a second timestamp part for storing a timestamp when said digital ticket was punched and said step of copying said digital ticket from said second repository for a fee is further comprised of the step of said second repository marking said first timestamp part of said digital ticket with a copy timestamp indicating the time and date when the copy of said digital ticket was made.

8. A method for controlling the number of times that a usage right attached to a digital work may be exercised in a computer controlled system for the distribution of digital works, said method comprising the steps of:

a) creating a digital work,
b) defining a usage right for said digital work, said usage right specifying a digital ticket indicating a predetermined number of times that said usage right may be exercised;
c) creating said digital ticket with an indicator of said predetermined number;
d) storing said digital work, said usage right and said digital ticket in a first repository;

SDOCID: <EP 0715247A1 I >

e) a second repository transmitting a request to access said digital work to said first repository, said request specifying said usage right;
f) said first repository determining if said digital ticket for said usage right indicates that said usage right has been exercised said predetermined number of times;
g) if said digital ticket indicates that said usage right has been exercised said predetermined number of times, said first repository denying access to said digital work;
h) if said digital ticket indicates that said usage right has not been exercised said predetermined number of times, said first repository granting access to said digital work; and
i) said first repository punching said digital ticket to indicate an instance of exercising said usage right.

9. A method for controlling the access to digital works in a computer controlled system for the distribution of digital works, said method comprising the steps of:

a) creating a digital work and a digital ticket;
b) defining a usage right for said digital work, said usage right specifying said digital ticket and a special ticket agent for punching said digital ticket;
c) storing said digital work in a first repository;
d) distributing said digital ticket to a second repository;
e) said second repository transmitting a request to access said digital work to said first repository, said request specifying said usage right;
f) said first repository determining if said second repository has said digital ticket;
g) said second repository presenting said digital ticket to said first repository;
h) said first repository presenting said digital ticket to said special ticket agent;
i) said special ticket agent determining if said digital ticket is valid;
j) if said digital ticket is invalid, said special ticket agent causing said request to access to be denied;
k) if said digital ticket is valid, said special ticket agent punching said digital ticket; and
l) said first repository granting access to said digital work by said second repository.

10. A method for distribution of upgrades to digital works in a computer controlled system for the distribution of digital works, said method comprising the steps of:

a) creating a digital ticket, said digital ticket permitting copying of an upgrade digital work;
b) distributing said digital ticket and a corresponding digital work to a first repository for a fee;
c) creating said upgrade digital work, said upgrade digital work having a copying right for permitting copying of said upgrade digital work to possessors of said digital ticket;
d) storing said upgrade digital work in a second repository;
e) said first repository transmitting a request to copy said digital work to said second repository;
f) said second repository determining if said first repository has said digital ticket;
g) said first repository presenting said digital ticket to said second repository;
i) said second repository determining if said digital ticket is valid;
j) if said digital ticket is invalid, said second repository causing said request to copy to be denied;
k) if said digital ticket is valid, said second repository punching said digital ticket; and
l) said second repository transmitting said upgrade digital work to said first repository.

DOCID: <EP_____0715247A1_L>

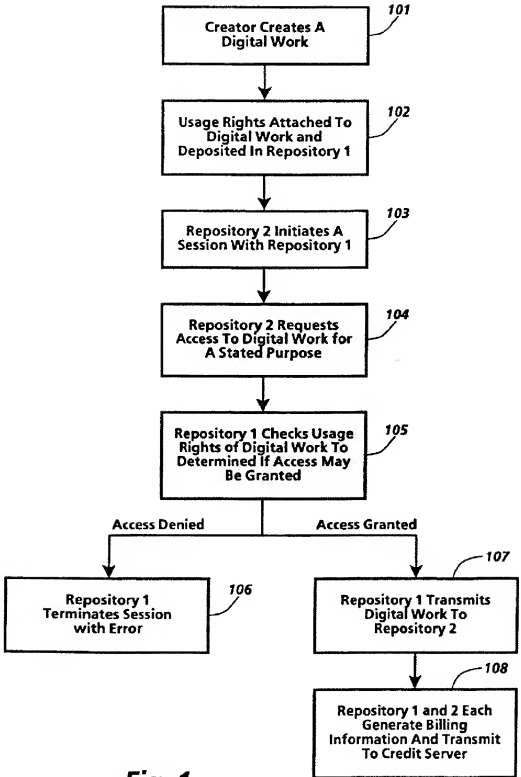

Fig. 1

SOCCID: <EP _____ 0715247A1_I_>

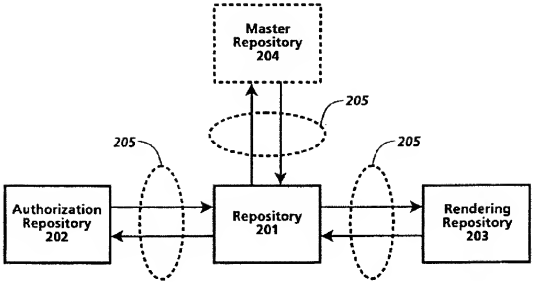

Fig. 2

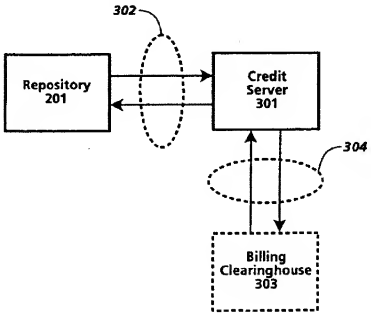

Fig. 3

SDOCID: <EP_____0715247A1_I_>

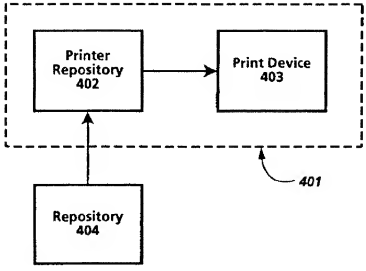

Fig. 4a

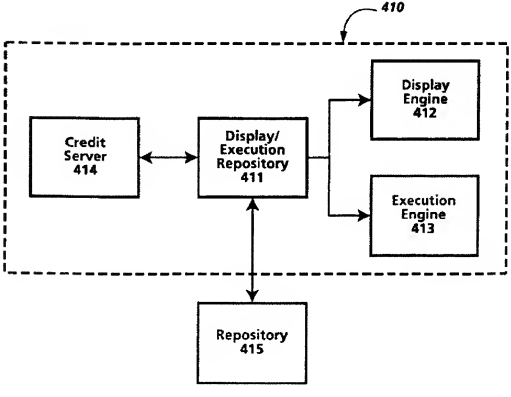

Fig. 4b

SOCOID <EP_____0715247A1_I_>

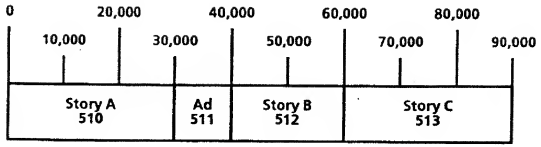

Fig. 5

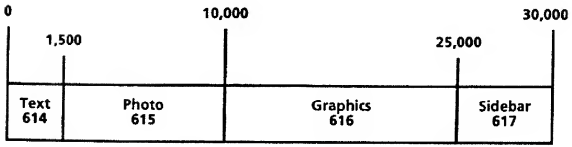

Fig. 6

DOCID <EP 0715247A1 I >

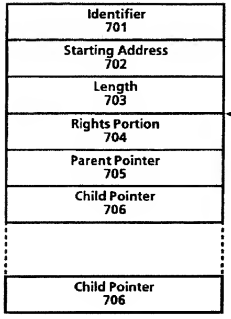

Fig. 7

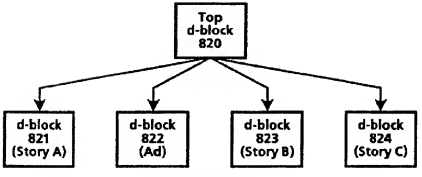

Fig. 8

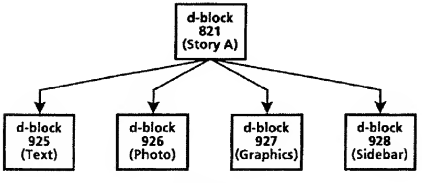

Fig. 9

SCOCID <EP_____0715247A1_I_>

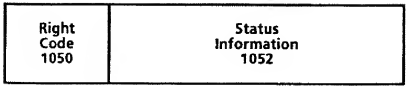

Fig.10

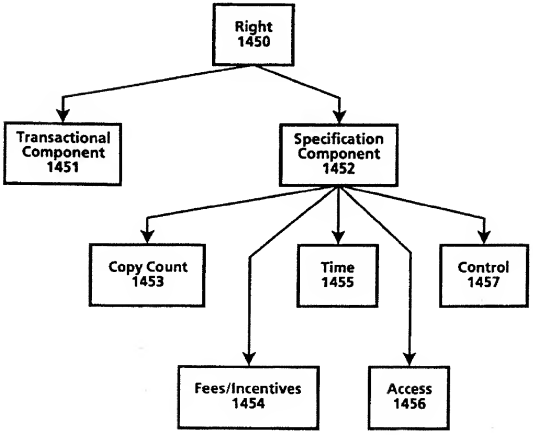

Fig.14

EDOCID: <EP ____0715247A1_I_>

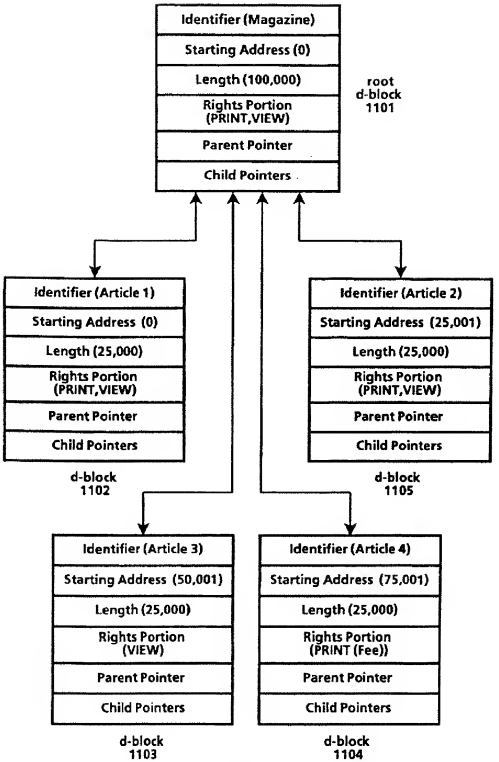

Fig.11

ISDOCID <EP_____0715247A1_I_>

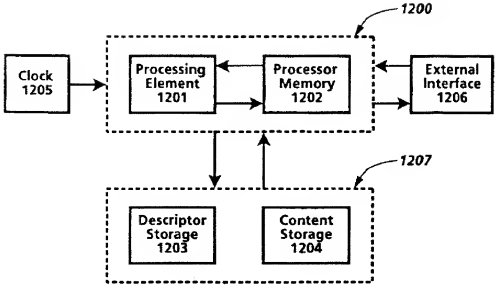
1200
Clock
1205
Processing
Element
1201
Processor
Memory
1202
External
Interface
1206
1207
Descriptor
Storage
1203
Content
Storage
1204

Fig.12

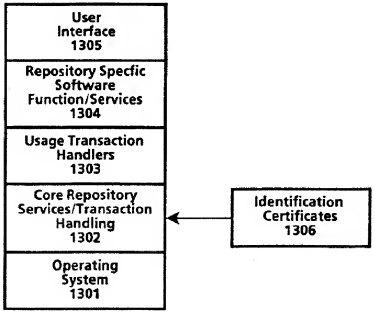
User
Interface
1305
Repository Specfic
Software
Function/Services
1304
Usage Transaction
Handlers
1303
Core Repository
Services/Transaction
Handling
1302
Operating
System
1301
Identification
Certificates
1306

Fig.13

```
1501 —Digital Work Rights: = (Rights*)
1502 —Right : = (Right-Code {Copy-Count} {Control-Spec} {Time-Spec }
        {Access-Spec} {Fee-Spec})
   1503 —Right-Code : = Render-Code | Transport-Code | File-Management-
          Code| Derivative-Works- Code | Configuration-Code
      1504 —Render-Code : = [ Play : {Player: Player-ID} | Print: {Printer: Printer-ID}]
      1505 —Transport-Code : = [Copy | Transfer | Loan {Remaining-Rights:
            Next-Set-of-Rights}]{(Next-Copy-Rights: Next-Set-of-Rights)}
      1506 —File-Management-Code  := Backup {Back-Up-Copy-Rights:
                                    Next-Set-of-Rights} | Restore | Delete | Folder
                                    | Directory {Name: Hide-Local | Hide-Remote}
                                    {Parts: Hide-Local | Hide-Remote}
      1507 —Derivative-Works-Code : =  {Extract | Embed | Edit{Process:
                                    Process-ID}} {Next-Copy-Rights :
                                    Next-Set-of Rights}
      1508 —Configuration-Code : = Install | Uninstall
      1509 —Next-Set-of-Rights : = {(Add: Set-Of-Rights)} {(Delete:
            Set-Of-Rights)} {(Replace: Set-Of-Rights )}{(Keep: Set-Of-Rights )}
1510 —Copy-Count : = (Copies:positive-integer | 0 | Unlimited)
1511 —Control-Spec : = (Control: {Restrictable | Unrestrictable}
                    {Unchargeable | Chargeable})
1512 —Time-Spec : = ({Fixed-Interval  | Sliding-Interval | Meter-Time}
                Until: Expiration-Date)
   1513 —Fixed-Interval : = From: Start-Time
   1514 —Sliding-Interval : = Interval: Use-Duration
   1515 —Meter-Time: = Time-Remaining: Remaining-Use
1516 —Access-Spec : = ([SC: Security-Class} [Authorization: Authorization-ID*}
        {Other-Authorization: Authorization-ID*} {Ticket: Ticket-ID})
1517 —Fee-Spec: = {Scheduled-Discount} Regular-Fee-Spec | Scheduled-Fee-Spec |
                Markup-Spec
   1518 —Scheduled-Discount: = Scheduled-Discount: (Scheduled-Discount:
                            (Time-Spec Percentage)*)
   1519 —Regular-Fee-Spec : = ({Fee: | Incentive: } [Per-Use-Spec | Metered-Rate-
                            Spec | Best-Price-Spec | Call-For-Price-Spec]
                            {Min: Money-Unit Per: Time-Spec}{Max:
                            Money-Unit Per: Time-Spec} To: Account-ID)
      1520 —Per-Use-Spec: = Per-Use: Money-unit
      1521 —Metered-Rate-Spec : = Metered: Money-Unit Per: Time-Spec
      1522 —Best-Price-Spec : = Best-Price: Money-unit Max: Money-unit
      1523 —Call-For-Price-Spec : = Call-For -Price
   1524 —Scheduled-Fee-Spec: = (Schedule: (Time-Spec Regular-Fee-Spec)* )
   1525 —Markup-Spec: = Markup: percentage To: Account-ID
```

Fig.15

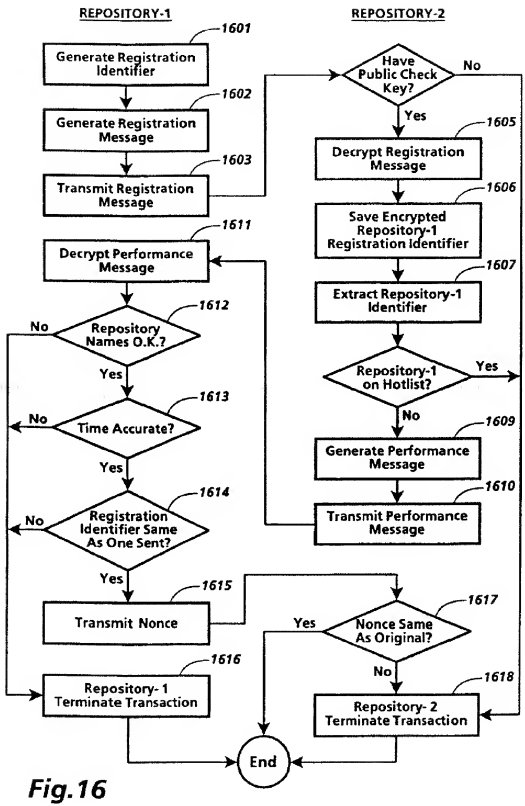

Fig.16

DOCID: <EP_____0715247A1_I_>

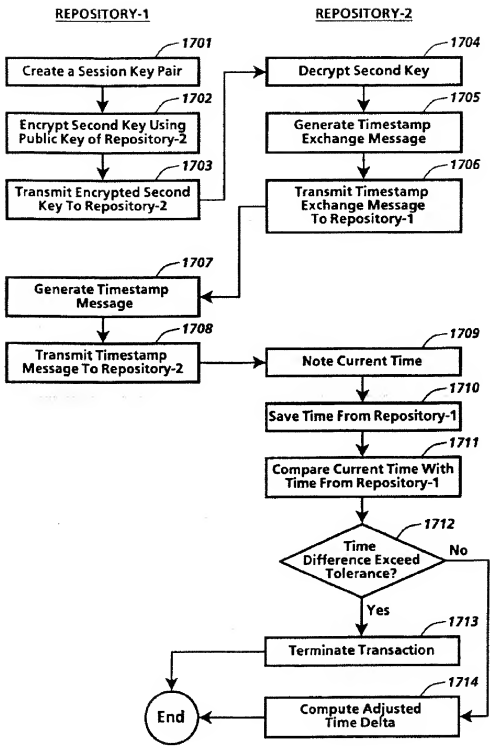
REPOSITORY-1
REPOSITORY-2
1701
Create a Session Key Pair
1702
Encrypt Second Key Using Public Key of Repository-2
1703
Transmit Encrypted Second Key To Repository-2
1704
Decrypt Second Key
1705
Generate Timestamp Exchange Message
1706
Transmit Timestamp Exchange Message To Repository-1
1707
Generate Timestamp Message
1708
Transmit Timestamp Message To Repository-2
1709
Note Current Time
1710
Save Time From Repository-1
1711
Compare Current Time With Time From Repository-1
1712
Time Difference Exceed Tolerance?
No
Yes
1713
Terminate Transaction
1714
Compute Adjusted Time Delta
End

Fig.17

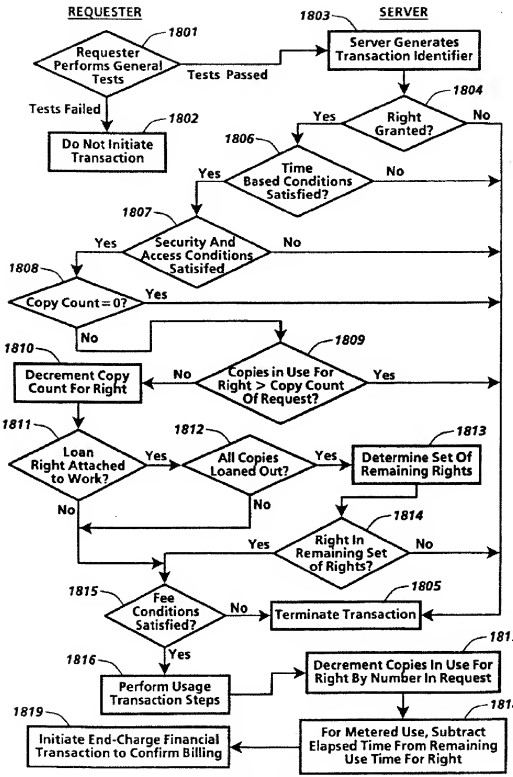

Fig.18

(800030 .EP 0715247A1 | >

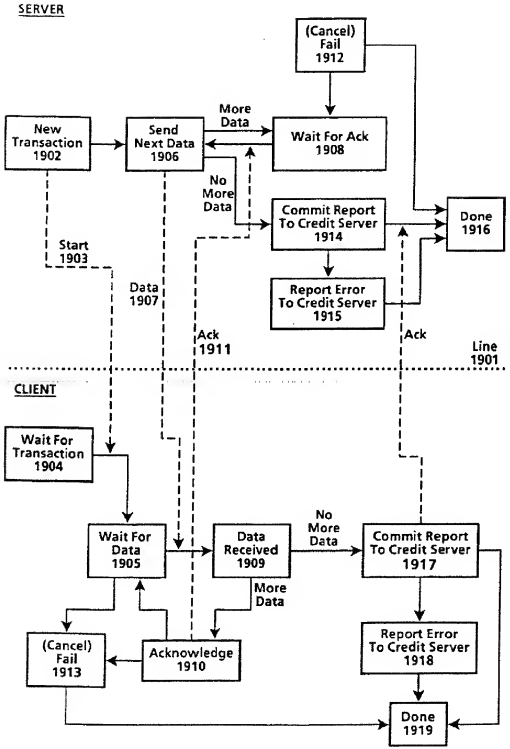

Fig.19

SDOCID <EP_____0715247A1_I_>

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 95 30 8422

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
A	WO-A-92 20022 (DIGITAL EQUIPMENT CORP.) * page 45, line 10 - page 64, line 17 * ---	1,5,8-10	G06F1/00
A	GB-A-2 236 604 (SUN MICROSYSTEMS INC) * page 9, line 11 - page 20, line 15 * ---	1,5,8-10	
A	US-A-5 291 596 (MITA) * the whole document * -----	1,5,8-10	
			TECHNICAL FIELDS SEARCHED (Int.Cl.6)
			G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	1 April 1996	Moens, R

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

EPODOC <EP 0715247A1 | >